

# User Story :

## L'histoire utilisateur au cœur

## du développement logiciel agile

### (partie 3 : Mettre en scène une histoire)



Isabelle BLASQUEZ  
@iblasquez

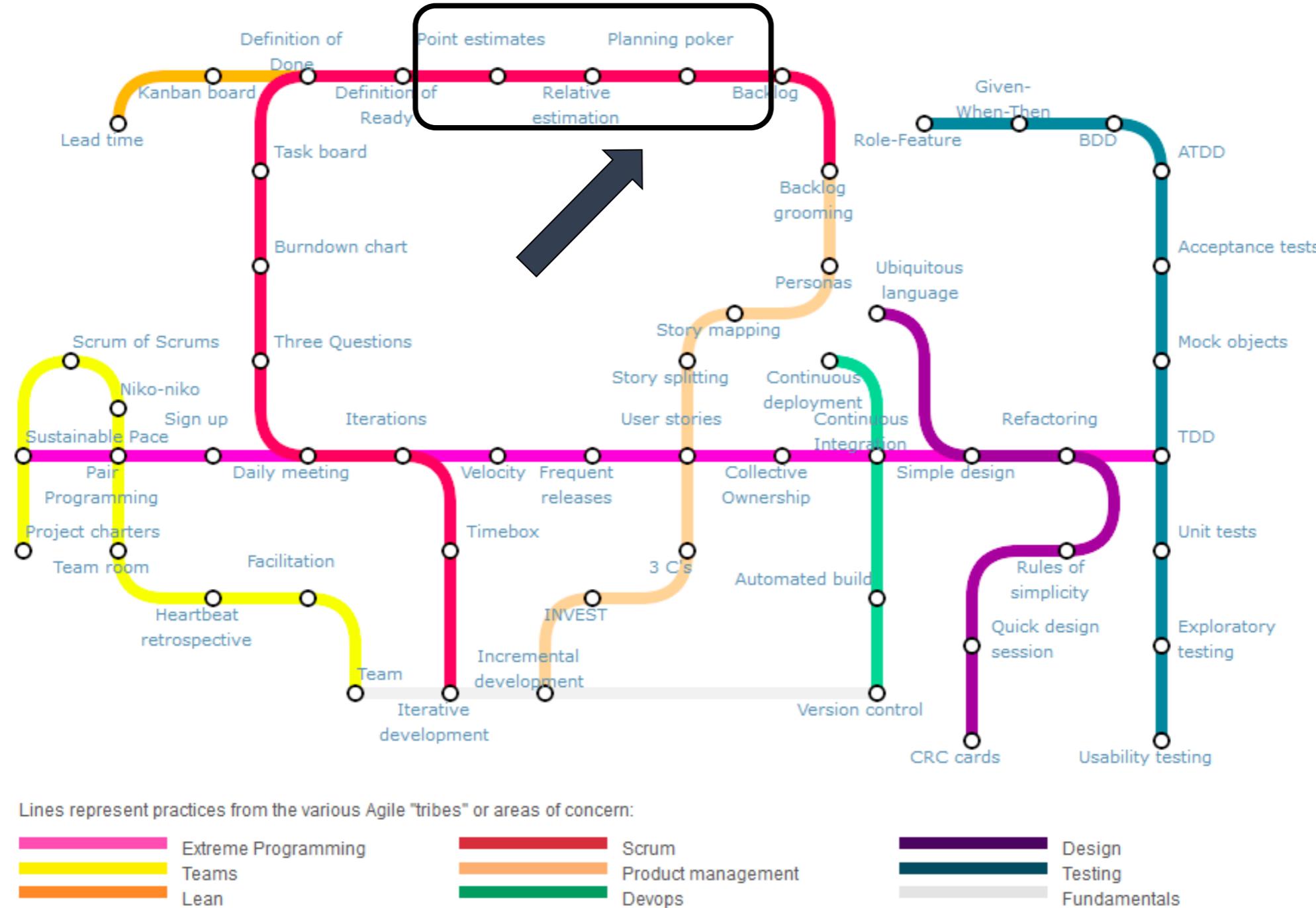
Septembre 2016

# **Mettre en scène une histoire**

# Prioriser & Estimer

*pour mieux détailler  
Et répartir la charge de travail*

# Zoom sur l'estimation des stories



Lines represent practices from the various Agile "tribes" or areas of concern:

Extreme Programming  
Teams  
Lean

Scrum  
Product management  
Devops

Design  
Testing  
Fundamentals

**Estimation = *effort* à fournir pour développer la story  
(*coût technique*) souvent exprimée en *Story Points***

**Etant que** recruteur

**Je veux** pouvoir effectuer  
le paiement d'une offre d'emploi  
avec un type de carte de crédit

**Afin de gagner du temps**

Note : carte Visa uniquement pour cette  
story

3

**Estimation relative  
en story points**

**Etant que** recruteur

**Je veux** pouvoir effectuer  
le paiement d'une offre d'emploi  
avec deux autres types de CB

**Afin de gagner du temps**

Note : Master Card & American Express

1

# Différentes méthodes d'estimation relative

Différentes méthodes basée sur la **comparaison**.

→ **Planning Poker** (story points)



→ **Tailles de T-shirt** (de XS à XXL)  
*qui seront ensuite convertis en story points*



1	2	3	5	8	13	20	!
x	x	x	x	x	x	x	x
	x	x	x	x	x		x
x		x	x	x	x		x
x		x	x				
x			x				
				x			
					x		
						x	
							x

→ Estimation **selon les couloirs**  
(Swinlane) : plus rapide que le Planning Poker

Extrait :  
<http://opatou.blogspot.fr/2012/01/ken-schwaber-co-fondateur-de-la-methode.html>  
Voir :  
[https://fr.wikipedia.org/wiki/Planning\\_poker](https://fr.wikipedia.org/wiki/Planning_poker)

Extrait :  
<http://rules.ssw.com.au/Management/RulesToBetterScrumUsingTFS/Pages/Do-You-Know-How-To-Size-Stories-Effectively.aspx>

Extrait :  
<http://theagilepirate.net/archives/109>

# Intérêt d'une réunion d'estimation

## Favorise la collaboration et les interactions

Dans Scrum

### ✓ Toute l'équipe participe à l'estimation

Le Product Owner présente, décrit et clarifie les User Stories

Le Scrum Master anime et gère le temps.

L'équipe de développement procède aux estimations de manière collaborative : l'équipe pose des questions, chaque membre vote, jusqu'à la recherche d'un consensus.



### ✓ Débat constructif

- Fait remonter les désaccords,
- Force à justifier les hypothèses,
- Permet à l'équipe d'avoir une grille d'évaluation commune de l'effort nécessaire à la réalisation d'une tache.

... et la discussion permet de mieux cerner les détails de la story  
⇒ Contribue à rendre les stories INVEST

# Prioriser

**Etant que** recruteur

**Je veux** pouvoir effectuer  
le paiement d'une offre d'emploi  
avec un type de carte de crédit

**Afin de gagner du temps**

Note : carte Visa uniquement pour cette story

**3**

**R1**

**Etant que** recruteur

**Je veux** pouvoir effectuer  
le paiement d'une offre d'emploi  
avec deux autres types de CB

**Afin de gagner du temps**

Note : Master Card & American Express

**1**

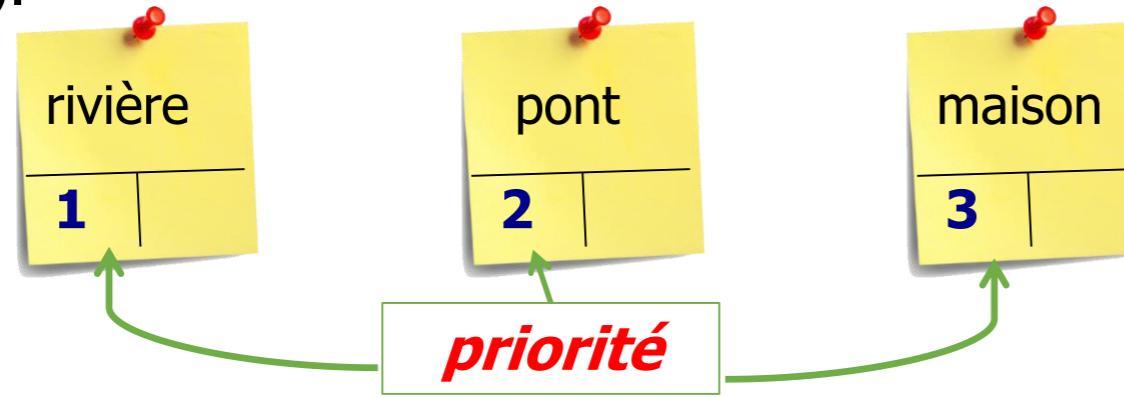
**R2**

C'est le Product Owner  
qui priorise les stories selon la valeur métier

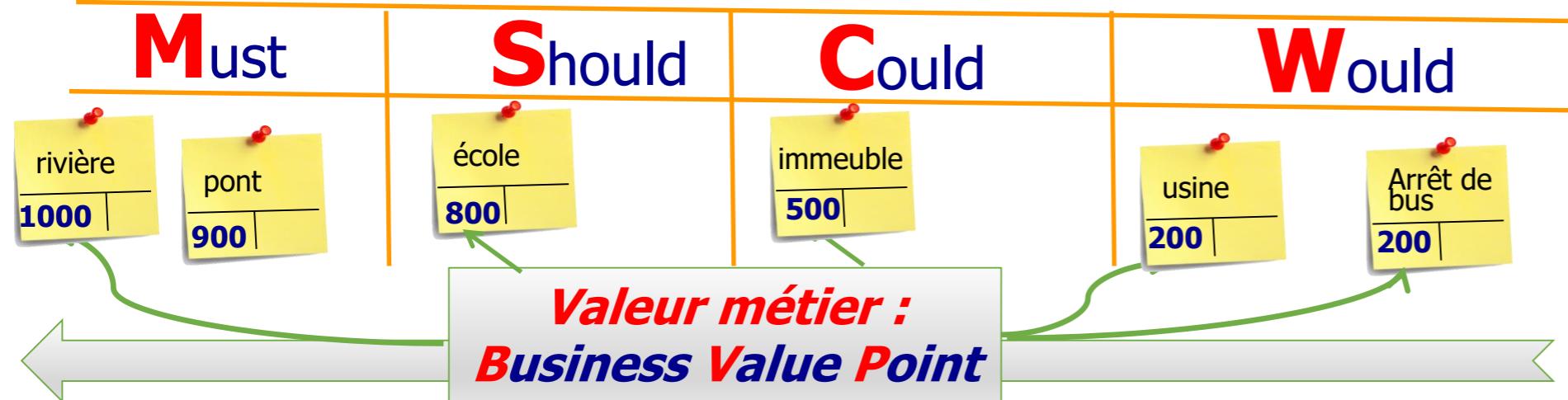
# Priorisation du Backlog

Dans Scrum, C'est le Product Owner qui **priorise les fonctionnalités** selon leur valeur métier.

- ✓ **Le plus simple** : Le **PO** ordonne lui-même son backlog par priorité : il trie les stories selon leur utilité (utilité ordinaire).

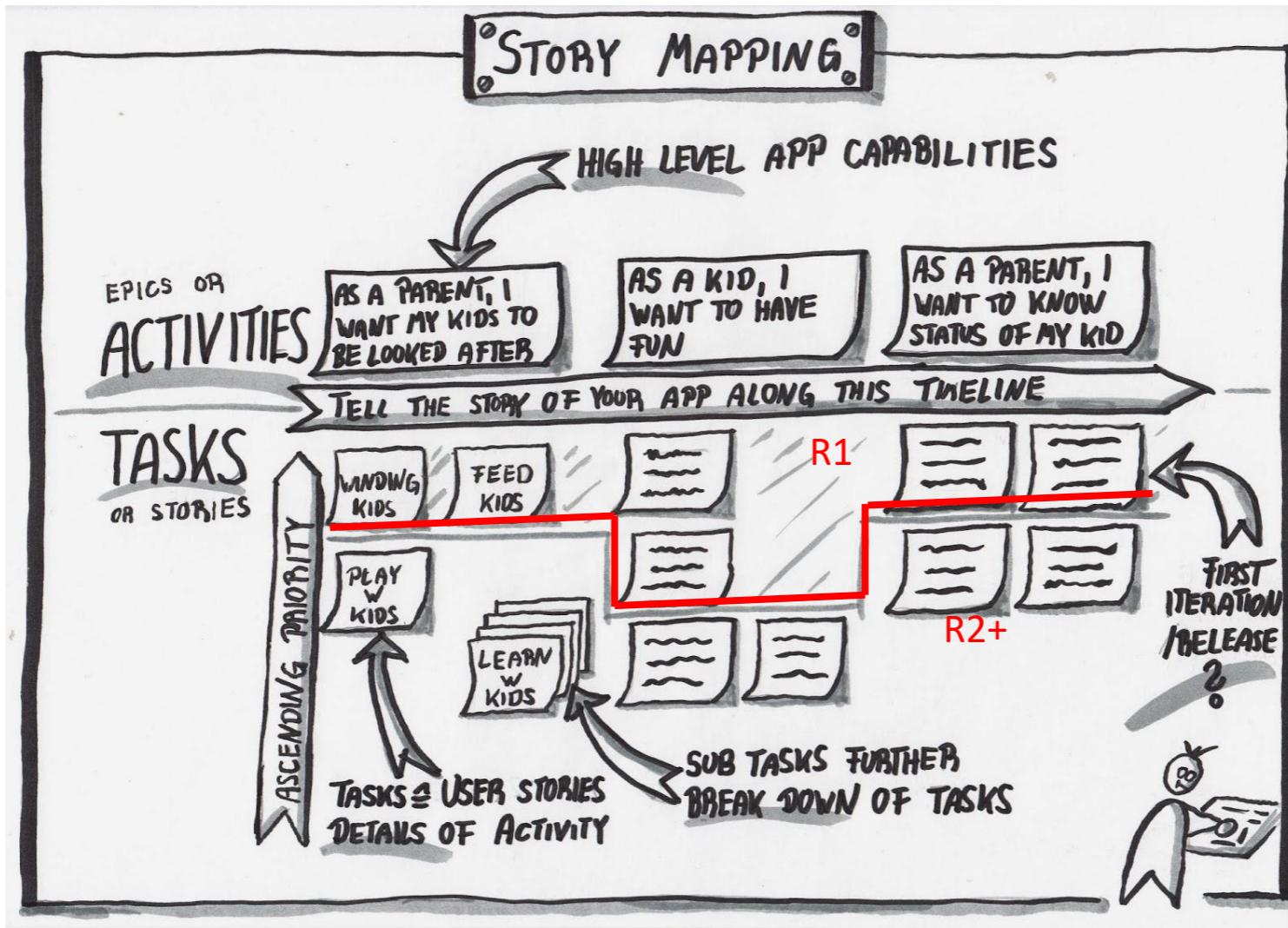


- ✓ **Utilisation d'une méthode de priorisation** la méthode **MoSCoW** permettant aussi de donner une valeur métier aux Stories



**Au-delà de la  
priorisation,  
la planification de  
release ...**

# Priorisation & Planification de release possible à partir du story mapping



Extrait : <http://itscertainlyuncertain.blogspot.de/2013/12/story-mapping-on-one-page.html>

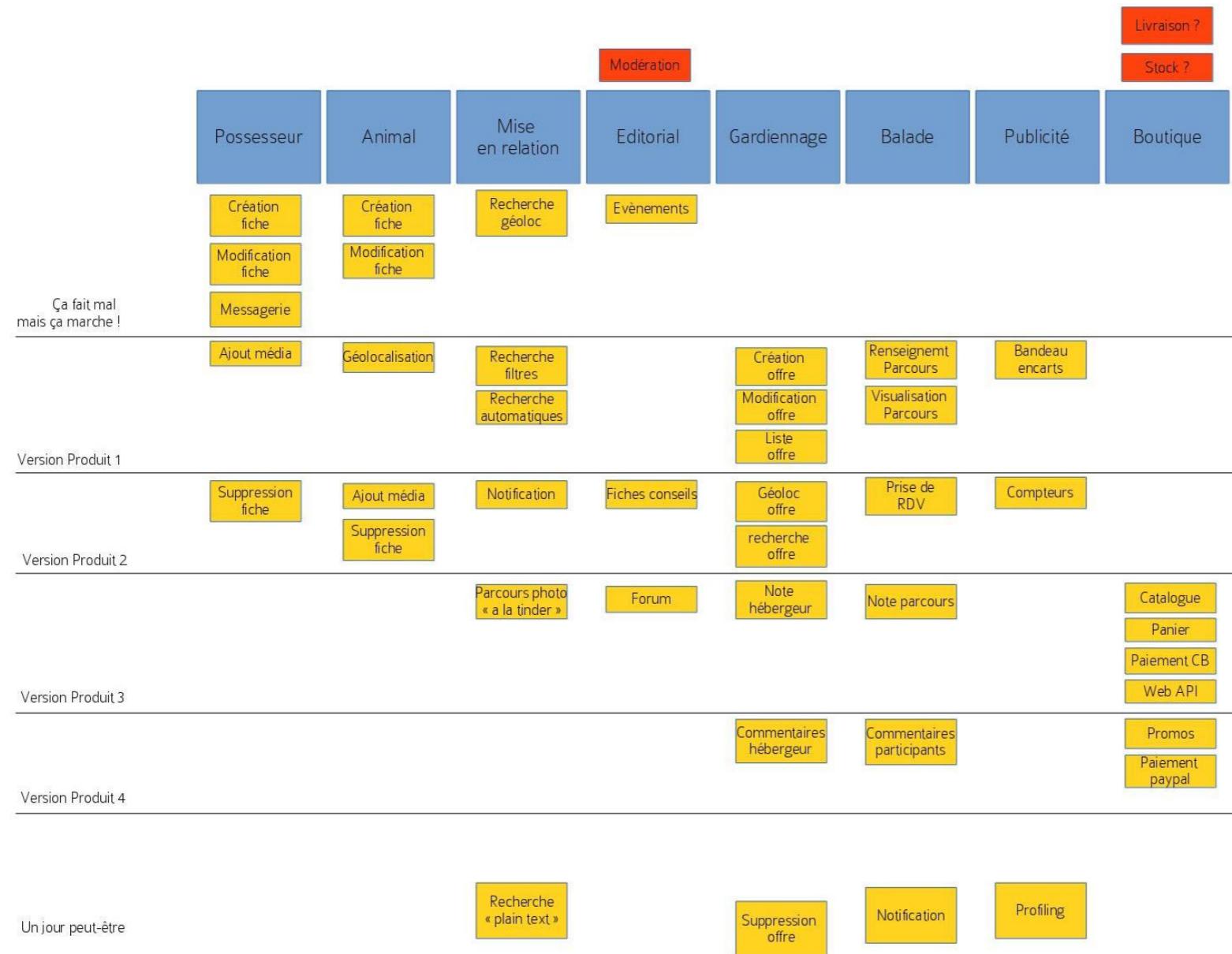
En savoir plus sur le story mapping :

[http://www.agileproductdesign.com/blog/the\\_new\\_backlog.html](http://www.agileproductdesign.com/blog/the_new_backlog.html)

<http://www.les-traducteurs-agiles.org/user-stories/pratique/2015/10/12/cartographies-des-user-stories.html>

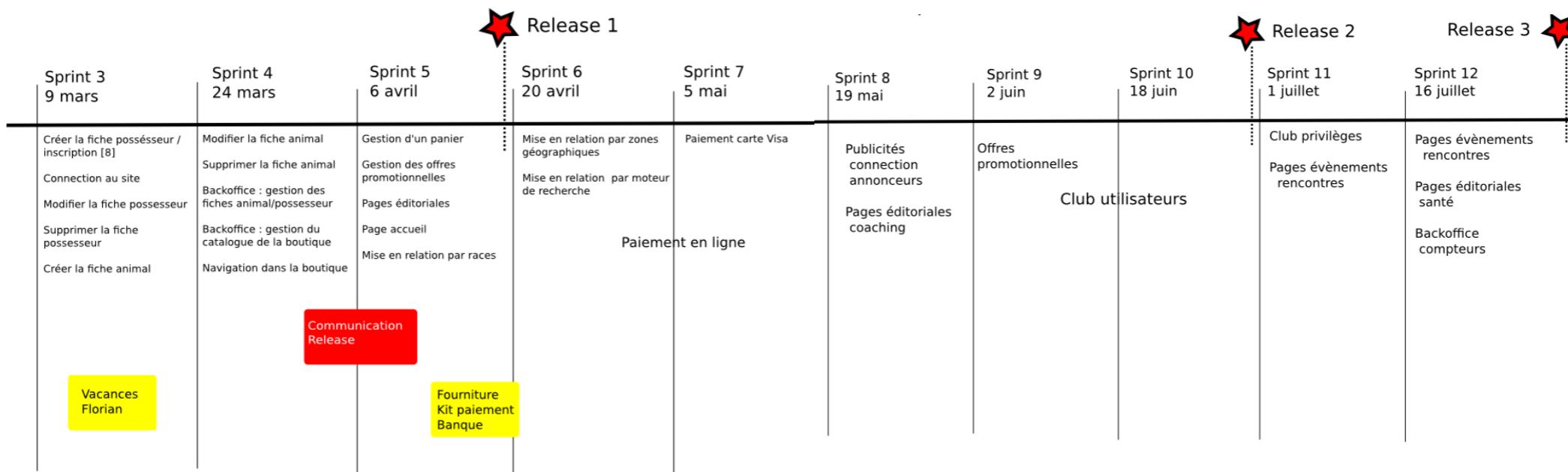
<http://www.infoq.com/fr/presentations/art-management-exigences-agiles>

# Exemple de priorisation à partir du story mapping



# Au delà de la priorisation, un planning de release...

## *Exemple de plan de Release de Peetic*



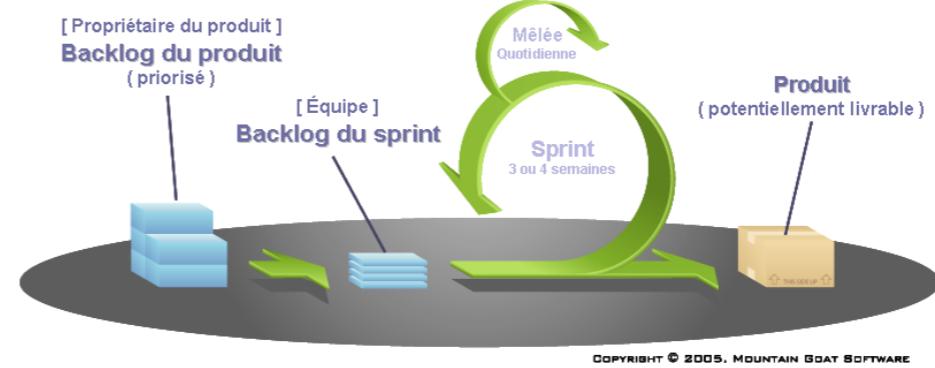
Extrait : <https://github.com/pablopernot/peetic>

**A noter : Une release est une série de sprints**



*A-t-on vraiment besoin de prévoir à moyen terme ?  
Sur quel horizon ? Avec quelle précision ?*

**3 Cérémonies**  
**Sprint Planning**  
Daily Scrum  
Sprint Review



# Quelques mots sur le découpage des stories en tâches ...

# Pourquoi découper en tâches ?

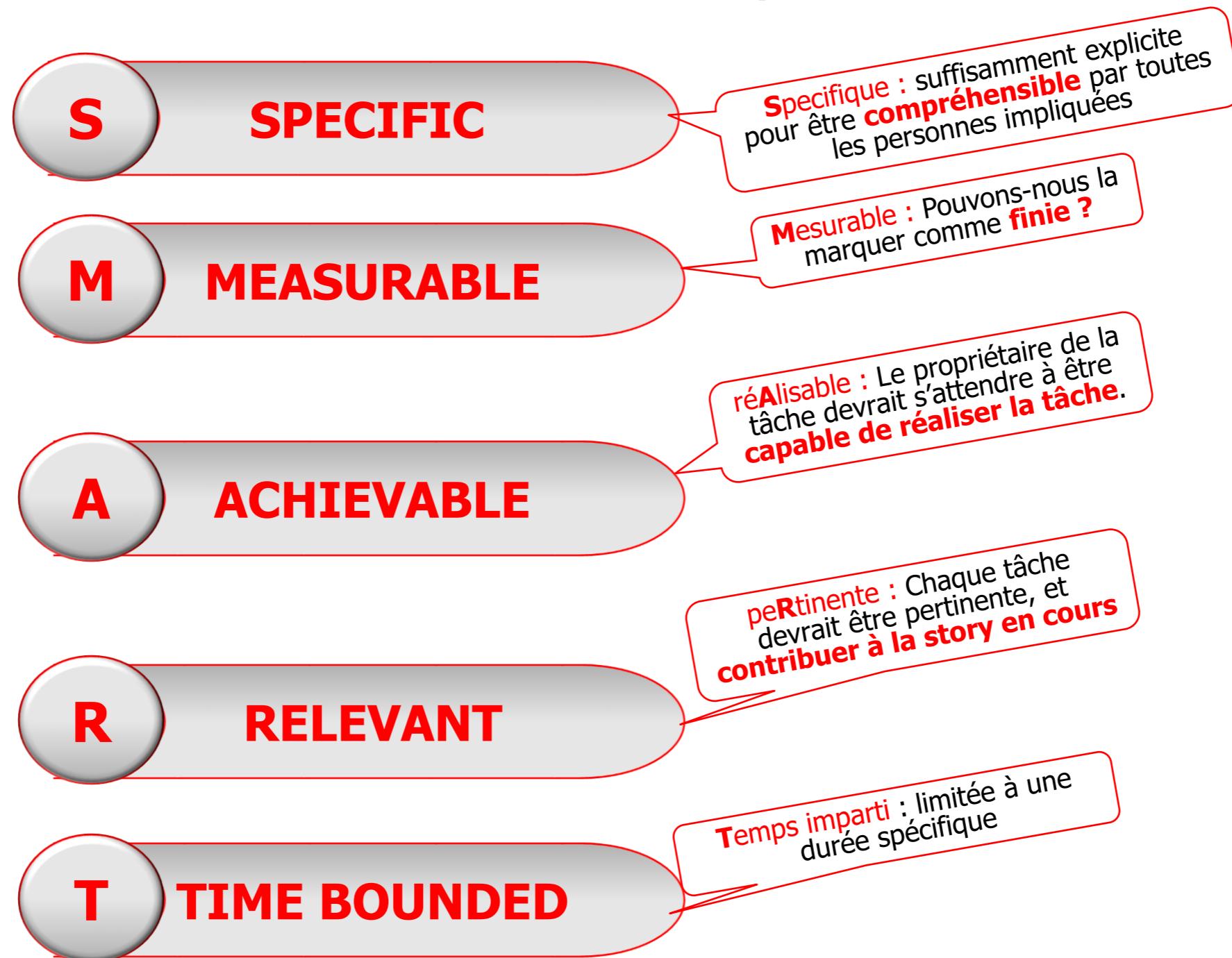
The sprint backlog is a **list of *tasks* identified by the *Scrum team*** to be completed during the Scrum sprint.

User Story	Tasks
As a member, I can read profiles of other members so that I can find someone to date.	Code the ...
	Design the ...
	Meet with Mary about ...
	Design the UI
	Automate tests ...
	Code the other ...
As a member, I can update my billing information.	Update security tests
	Design a solution to ...
	Write test plan
	Automate tests ...
	Code the ...

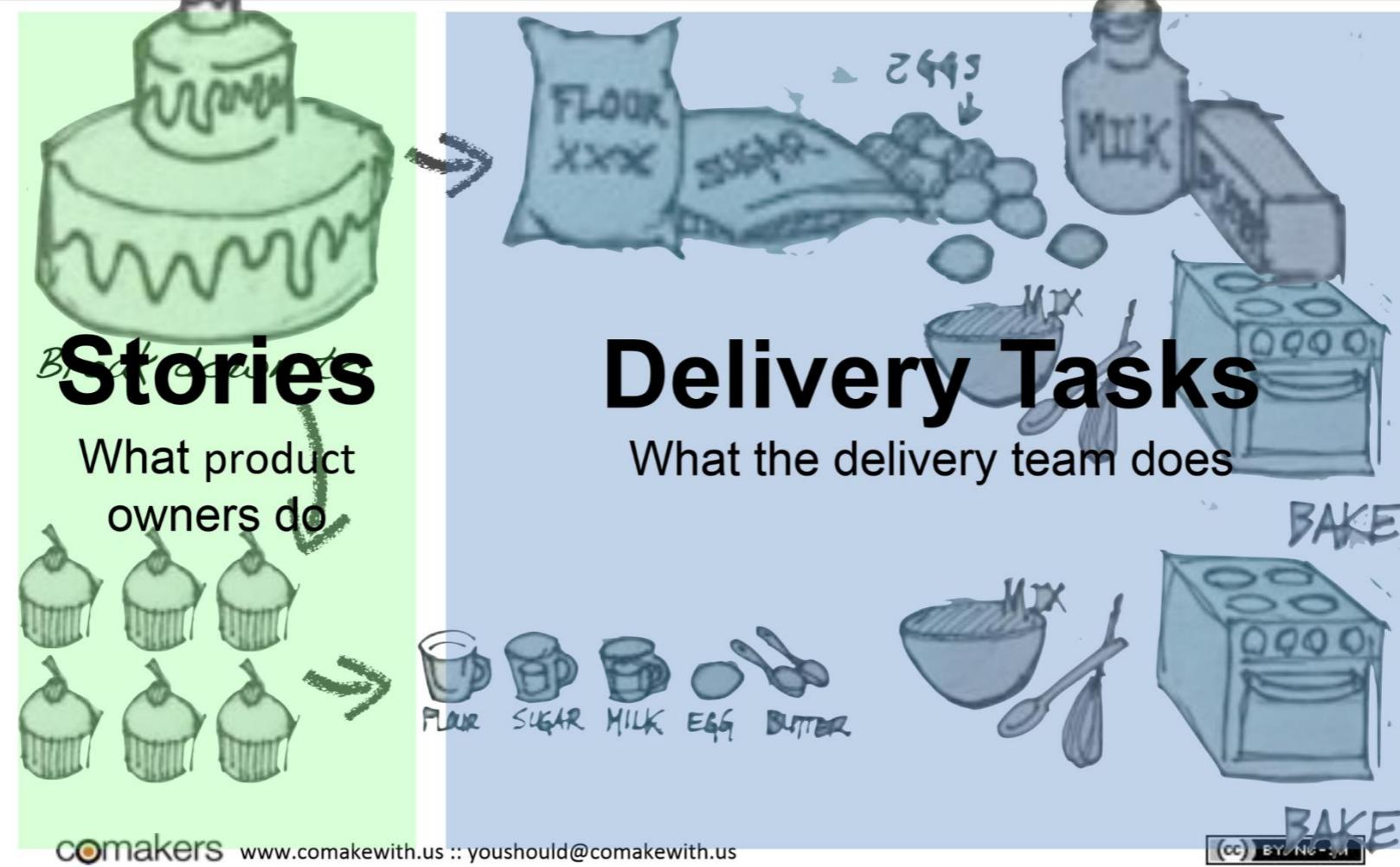
Extrait <https://www.mountaingoatsoftware.com/agile/scrum/sprint-backlog>

**Idéalement, une tâche ne devrait pas excéder une journée**

# INVESTir dans de bonnes stories, avec des tâches SMART



When breaking down stories, each story  
is a  deliverable, testable part

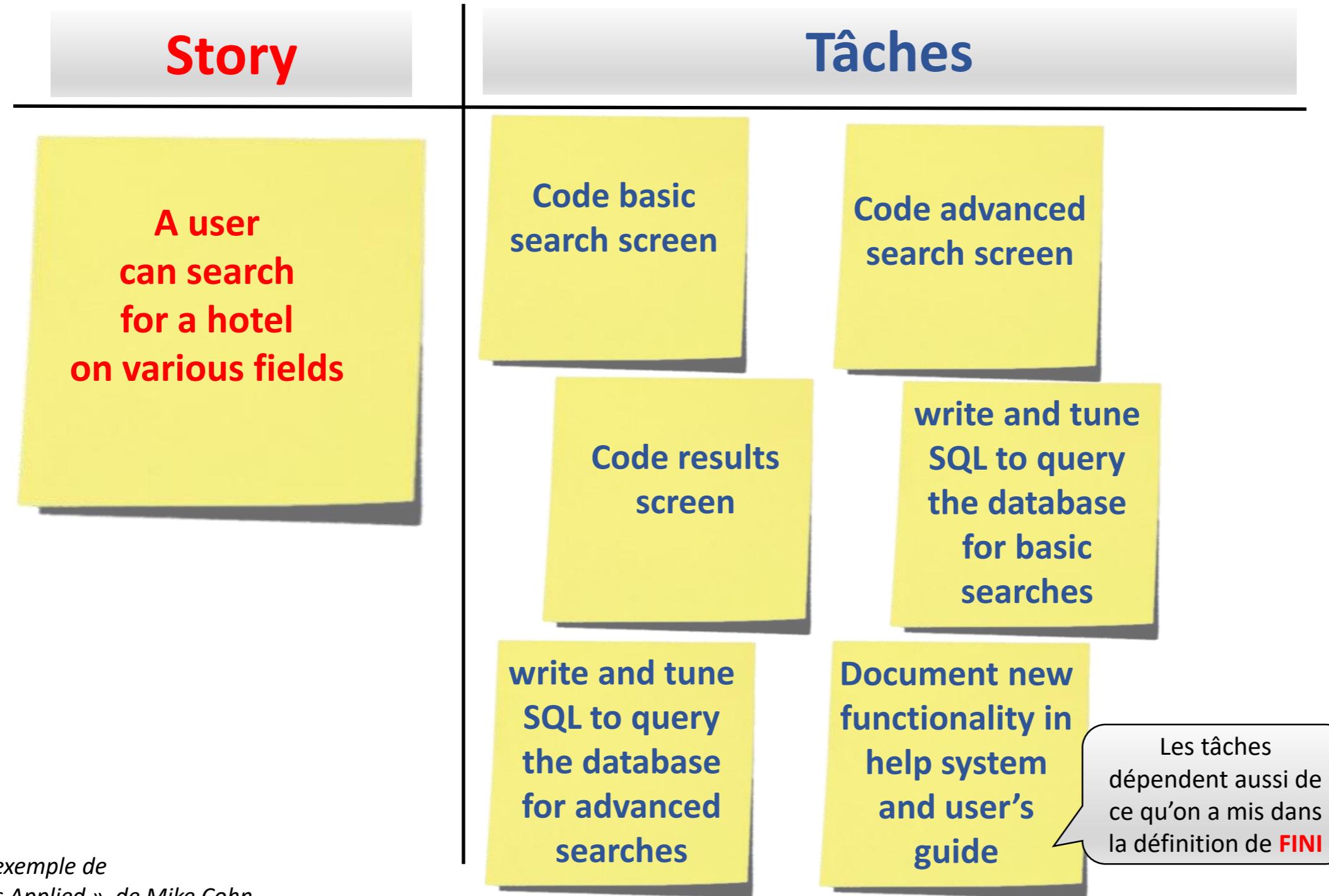


Extrait :

[https://submissions.agilealliance.org/system/attachments/attachments/000/000/093/original/  
Patton\\_Agile\\_Requirements\\_Product\\_Mgmt.pdf?1385085680](https://submissions.agilealliance.org/system/attachments/attachments/000/000/093/original/Patton_Agile_Requirements_Product_Mgmt.pdf?1385085680)

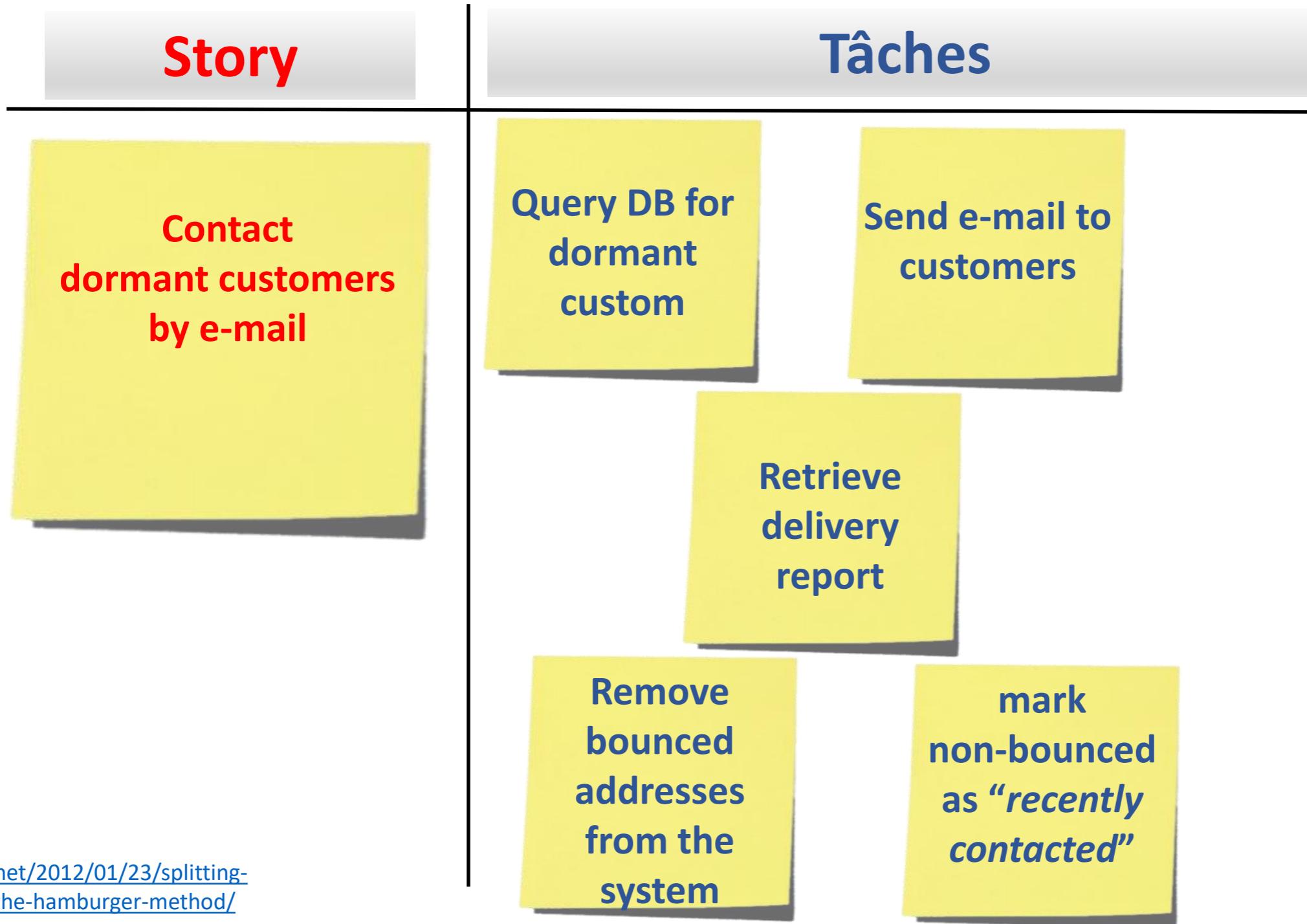
Isabelle BLASQUEZ - 2016

# Exemple de découpage d'une User story en tâches SMART



Les tâches dépendent aussi de ce qu'on a mis dans la définition de **FINI**

# Autre exemple de découpage en tâches SMART



Adapté de :

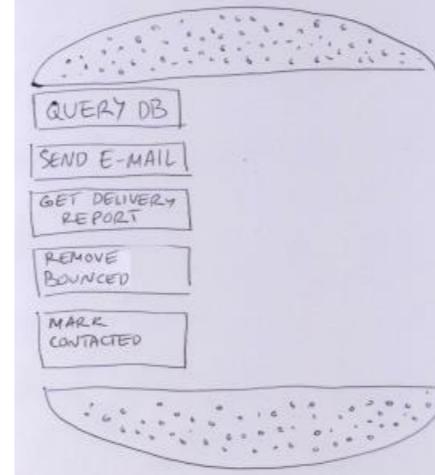
<http://gojko.net/2012/01/23/splitting-user-stories-the-hamburger-method/>

Isabelle BLASQUEZ - 2016

# Le découpage en tâches peut être plus ou moins fin

**Story**

Contact  
dormant  
customers  
by e-mail

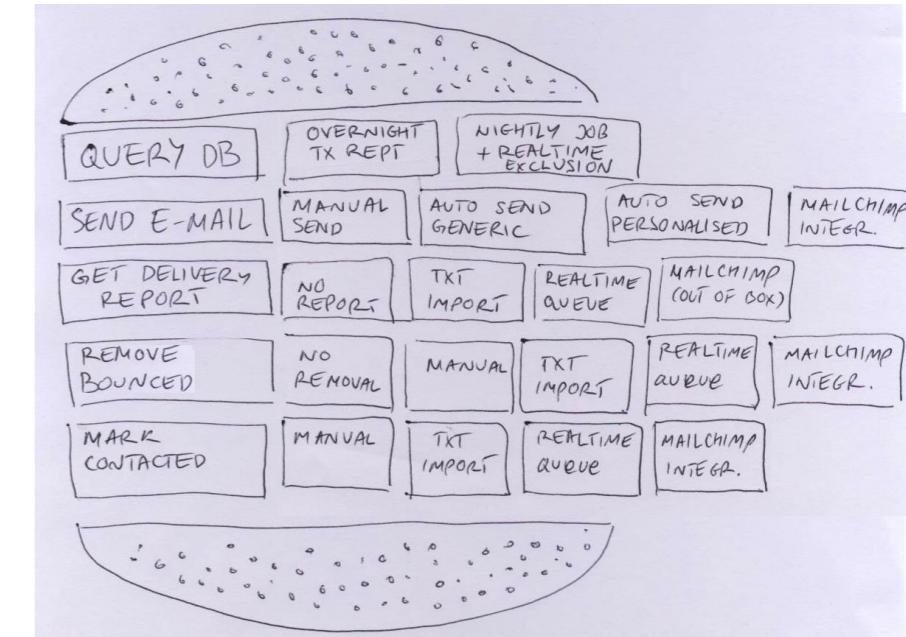


**Step 1:**  
**Identify tasks**

**Le découpage peut dépendre :**

- de la granularité souhaitée,
  - de la manière dont on souhaite découper les stories,
  - du niveau de connaissance dont on dispose sur la story
- c-a-d plus ou moins de précision sur les critères d'acceptation,  
qu'il faudra alors peut-être compléter...*
- ...

**Idéalement,  
une tâche ne  
devrait pas excéder  
une journée**

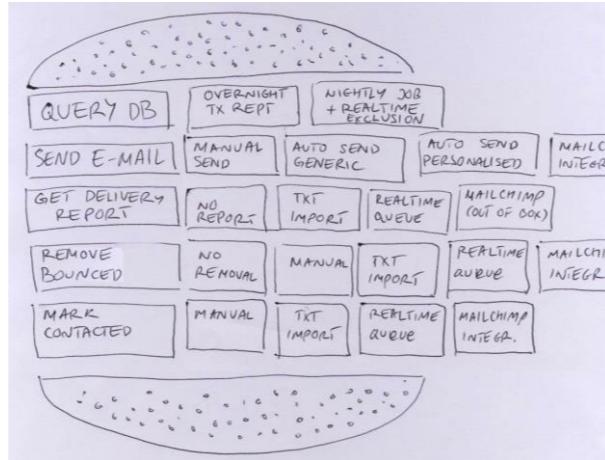
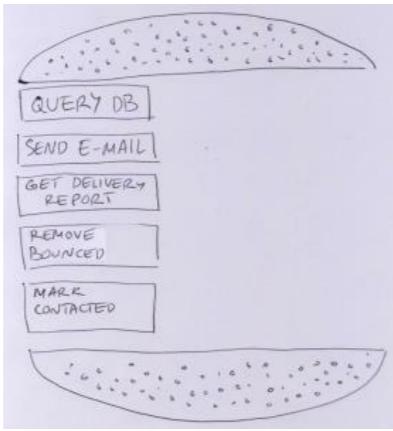


**Step 2:**  
**Identify options for tasks & Combine results**

# Et après... comment va-t-on choisir les tâches dans le sprint ? Exemple de la Méthode du hamburger...

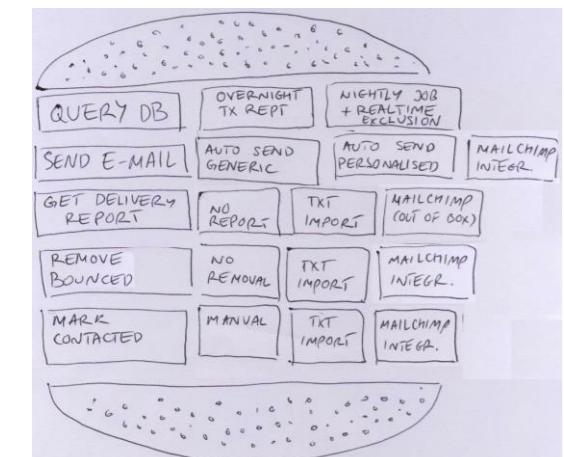
## Story

Contact dormant customers by e-mail

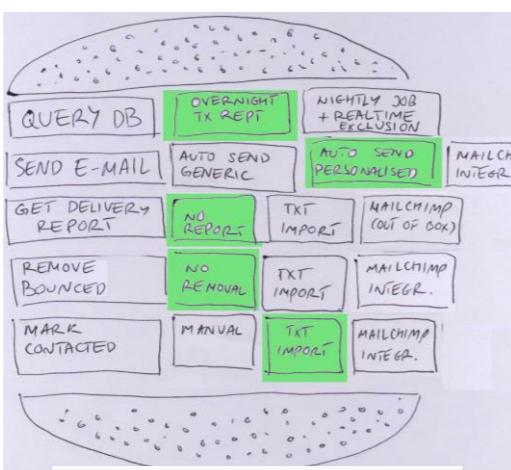


**Step 1:**  
Identify tasks

**Step 2 et 3 :**  
Identify options  
for tasks  
&  
Combine results



**Step 4:**  
Trim the hamburger

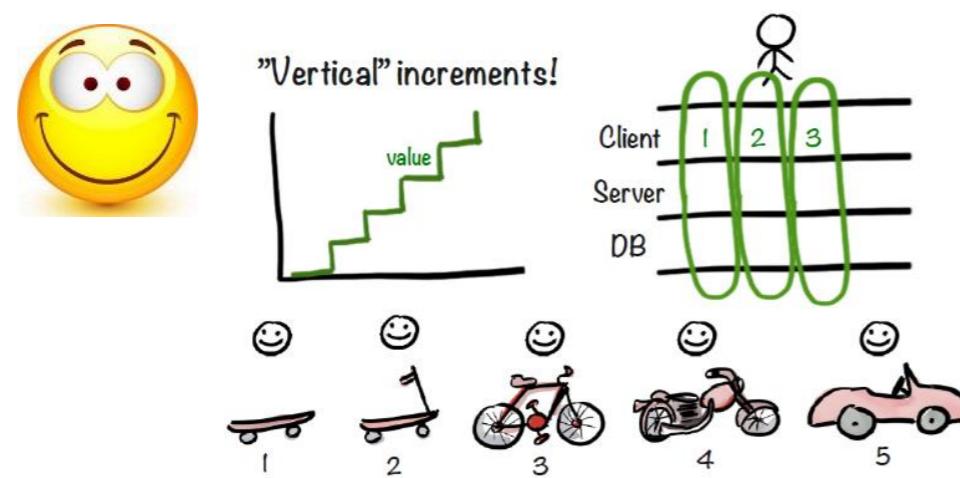


**Step 5:**  
Take the first bite

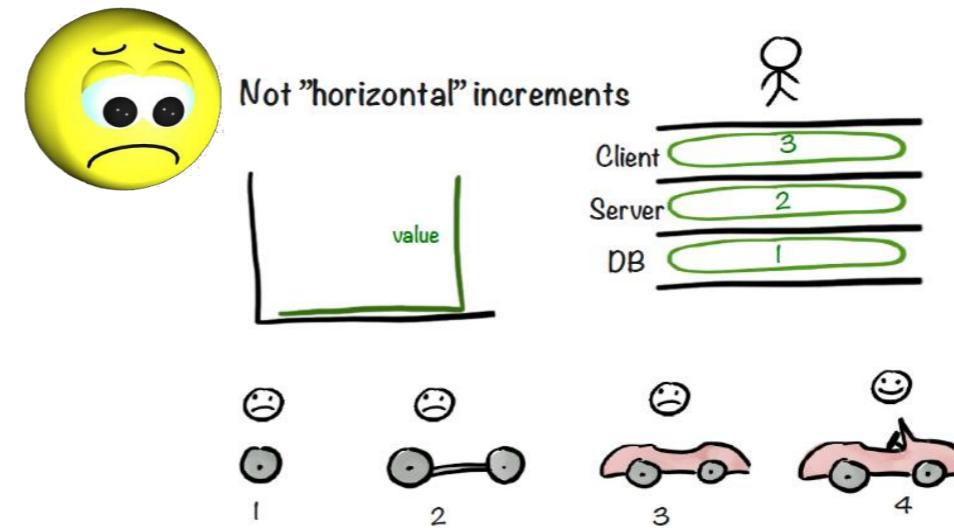
**Step 6:**  
Take another bite...  
and continue

Pour information

# Favoriser un découpage vertical ...



Henrik Kniberg



Henrik Kniberg

Illustrations sur <http://fr.slideshare.net/RichardPDoerer/what-isagile-henrik-kniberg-august-20-2013>

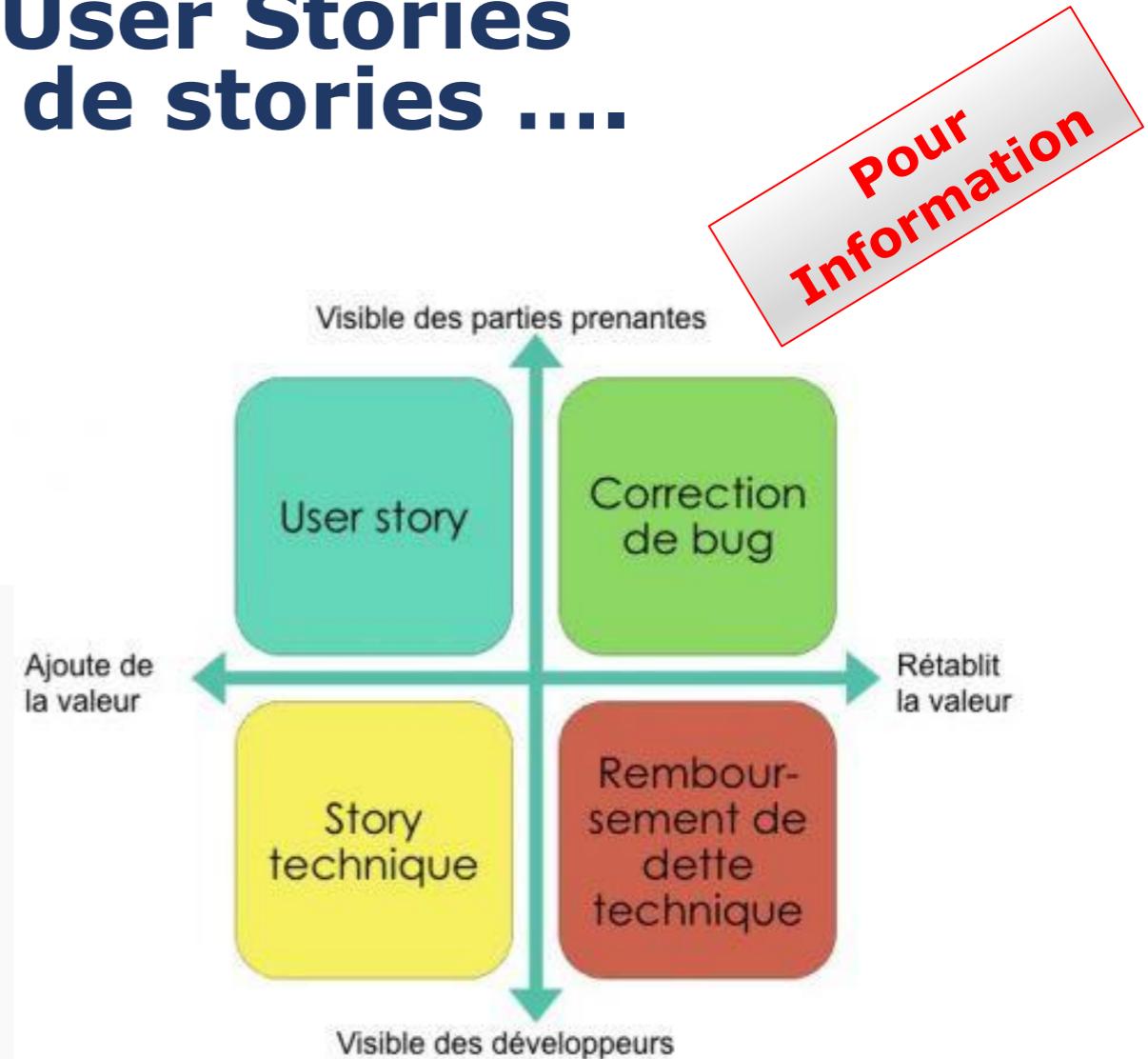
# Remarque : Dans le backlog de sprint, il pourra aussi y avoir des tâches qui ne seront pas forcément liées aux User Stories mais à d'autres types de stories ....

Attention, dans les user stories il y a toujours des travaux techniques (conception, code) et c'est très bien. Il ne s'agit pas de faire une story technique pour chaque user story, en y mettant les travaux techniques.

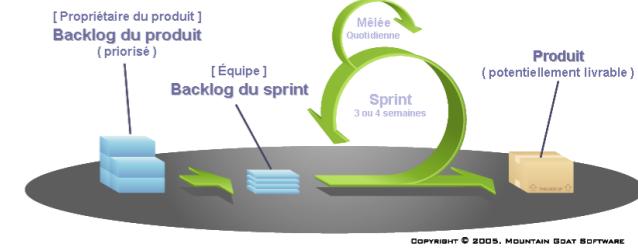
## Spike proposé par XP

La story "technique" se justifie dans plusieurs situations :

- il y a une décision technique à prendre sur la façon de réaliser une user story,
- il y a un risque technique qu'on veut lever avant la réalisation d'une user story,
- des travaux techniques, visant à améliorer la façon dont l'équipe développe ou la qualité : infrastructure, logistique, nouvel outil, formation, refactoring...



# A la fin de la réunion de planification de sprint, Préparation du *Task Board* par le Scrum Master



Le Scrum master crée le tableau des tâches à partir du **backlog de sprint**.  
Il le fait **après** la réunion de planification du sprint,  
mais **avant** la première mêlée quotidienne.

Story	Tâches			Done
	To Do	In Process	To Verify	
As a user, I... 8 points	Code the... 9 Code the... 2 Test the... 8	Test the... 8 Code the... 8 Test the... 4		
As a user, I... 5 points	Code the... 8 Code the... 4	Test the... 8 Code the... 6		

En savoir plus sur le task board : <https://www.mountaingoatsoftware.com/agile/scrum/task-boards>

Isabelle BLASQUEZ - 2016

# Amélioration possible du management visuel en utilisant des tâches en couleur ...

Les couleurs permettront de mieux mettre en évidence les interventions nécessaires pour finir une story.

Sprint 3 : début le 15/3, fin le 29/3			
	À faire	En cours	Fini
story	tâche tâche		
	tâche tâche tâche		
story	tâche tâche tâche		
	tâche tâche tâche		
story	tâche tâche tâche tâche		
	tâche tâche tâche		

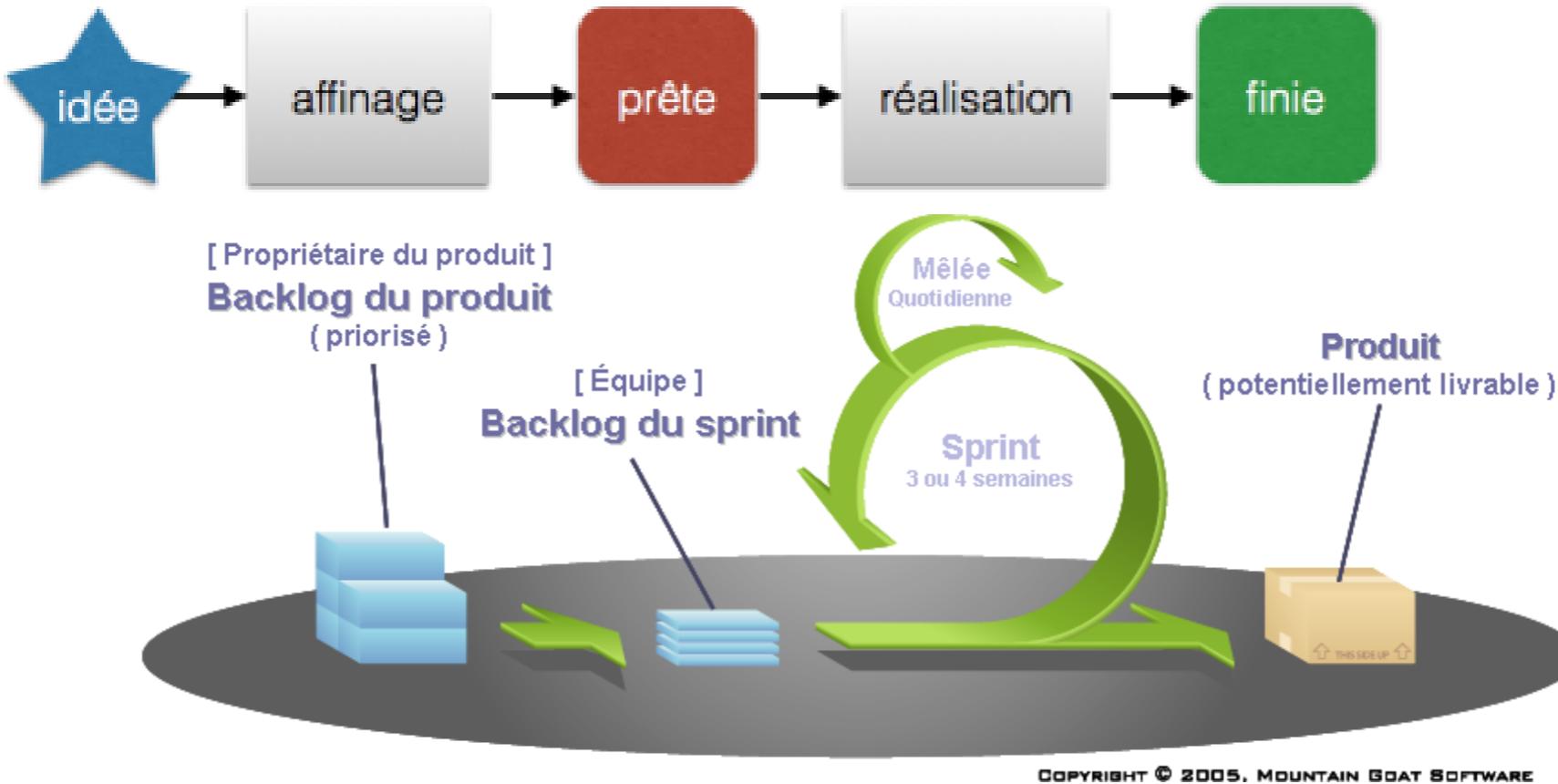
Pour  
Information

Exemple de code couleur possible :

- les tâches **de test en vert**,
- les tâches faites par une **personne non présente dans l'espace commun** et intervenant ponctuellement en **jaune**,
- les autres **tâches en rose**.

**Zoom sur  
le cycle de vie  
des stories ...**

# Workflow d'une story



On arrive à la réunion de planification du sprint avec une **liste de stories prêtées**. Ces stories sont déjà estimées : c'est généralement une condition pour les considérer comme prêtées.

Leur estimation, faite par exemple avec un Planning Poker, date au mieux de quelques jours (lors des revues de backlog du sprint) et probablement pour certaines de beaucoup plus.

# Zoom sur la définition de Prêt



# Définition de prêt

## Définition de "prêt"

Pratique

### De quoi s'agit-il?

Par analogie à la "définition de 'fini'", l'équipe explicite et rend visible les critères (généralement une déclinaison de la grille INVEST) faute desquels une fonctionnalité ne saurait faire l'objet d'un travail au cours de l'itération qui commence.

### On l'appelle également...

Traduction directe de "definition of ready".

### Quels bénéfices en attendre?

- évite de commencer à travailler alors que les critères de satisfaction ne sont pas clairs, ce qui risquerait d'entraîner de coûteux allers-retours avant de se mettre d'accord

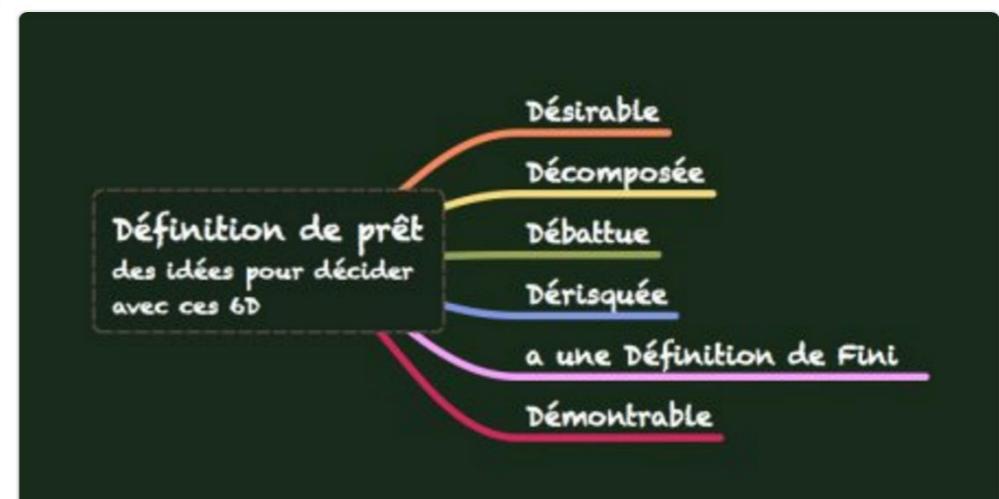
### Origines

- appellation très récente, formulée par Jeff Sutherland en 2008



Claude Aubry @claudeaubry · 3 juin

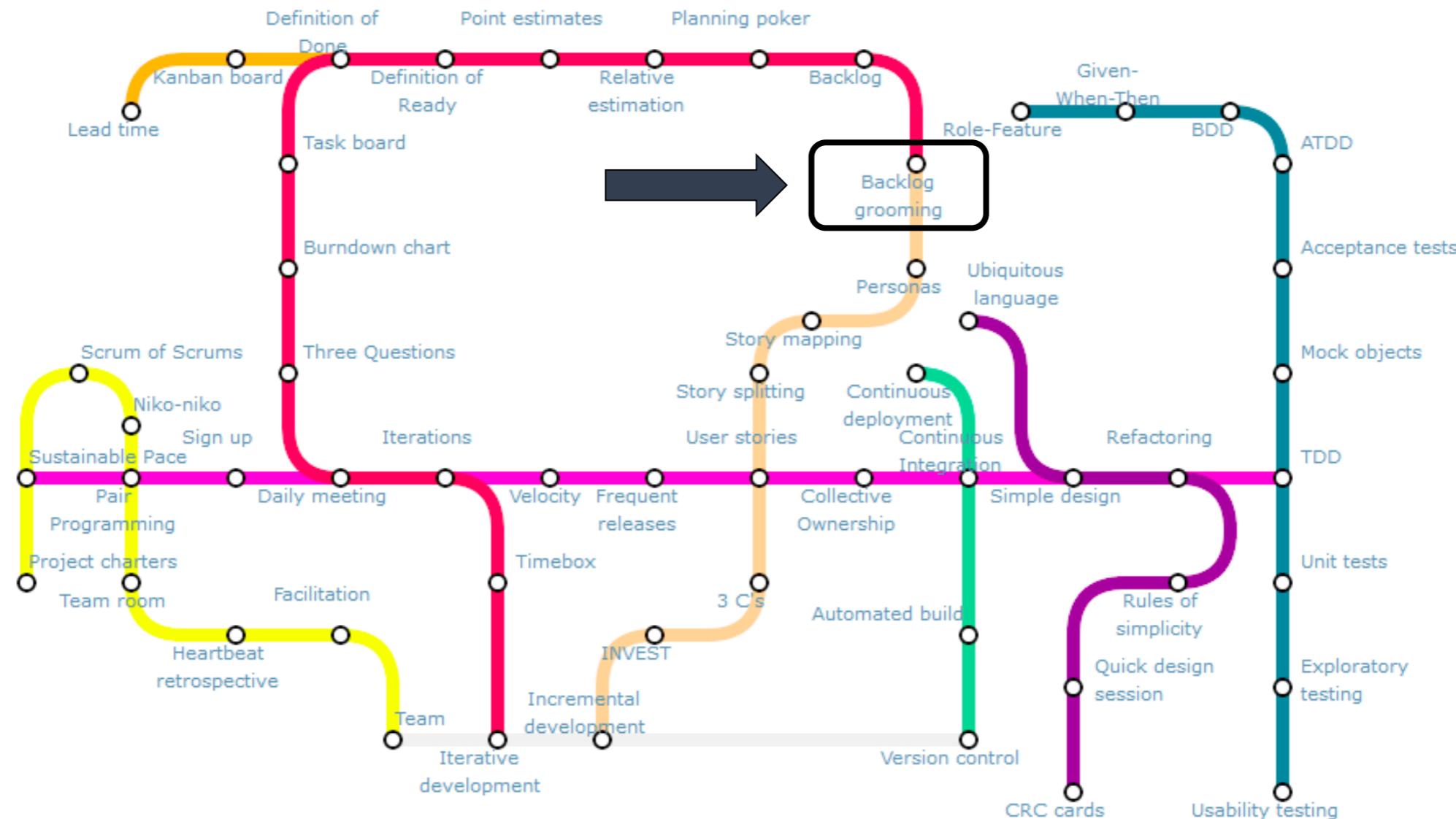
[scrum] 6 critères pour décider qu'une story est prête [buff.ly/1WyX8ZX](http://buff.ly/1WyX8ZX)



En savoir plus sur la proposition des 6 critères :

<http://www.aubryconseil.com/post/Decider-qu-une-story-est-prête>

# Et un mot sur le backlog Grooming pour terminer ...



Lines represent practices from the various Agile "tribes" or areas of concern:

Extreme Programming  
Teams  
Lean

Scrum  
Product management  
Devops

Design  
Testing  
Fundamentals

# Zoom sur le Backlog Grooming (revue de backlog ou affinage du backlog)

## Backlog Grooming

### Definition

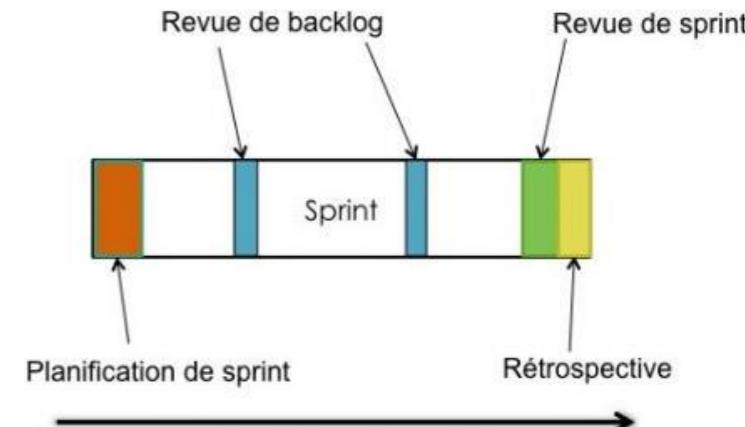
The team (or part of the team including the product owner) meet regularly to "groom the product backlog", in a formal or informal meeting which can lead to any of the following:

- removing user stories that no longer appear relevant
- creating new user stories in response to newly discovered needs
- re-assessing the relative priority of stories
- assigning estimates to stories which have yet to receive one
- correcting estimates in light of newly discovered information
- splitting user stories which are high priority but too coarse grained to fit in an upcoming iteration

### Also Known As

Other terms include "Story Time" (see timeline), "Backlog Refinement Meeting". The term reflects an organic approach to maintaining the backlog: the intended imagery is that of trimming, pruning, cleaning, as with a plant.

**La Revue de backlog** permet d'avancer collectivement sur le bac(log) de culture.

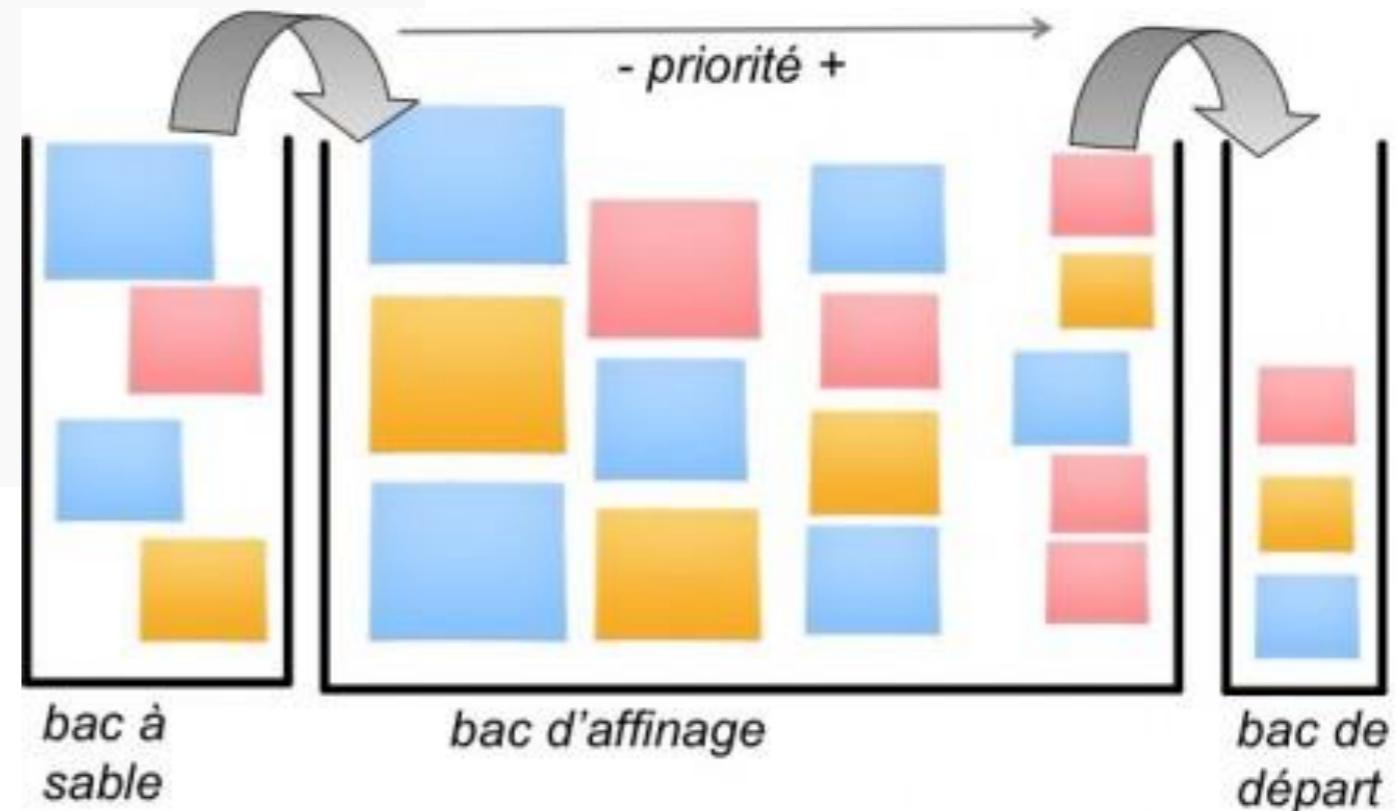


Extrait : <http://www.aubryconseil.com/post/Les-reunions-d-un-sprint>

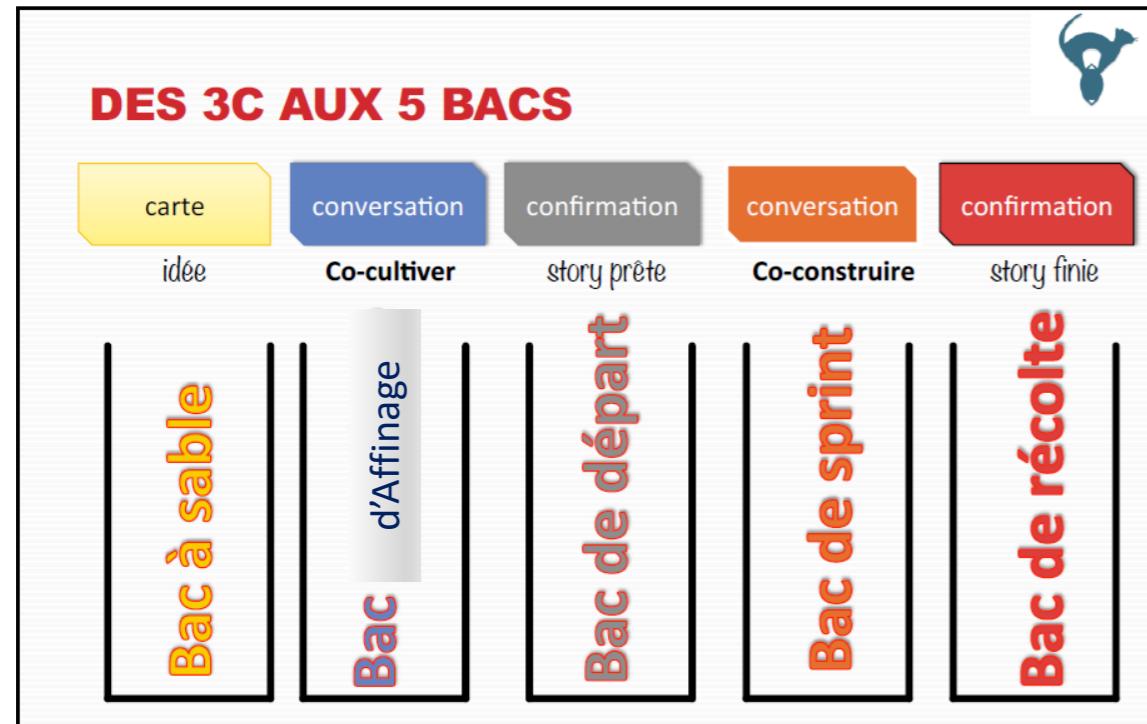
# En savoir plus sur les travaux d'affinage...

Les travaux d'affinage consistent en :

1. avoir une compréhension partagée de quelques stories pour qu'elles soient prêtes pour le sprint
2. revoir l'ordre des stories (les priorités)
3. décomposer des stories non élémentaires (epics)
4. estimer ce qui n'est pas encore estimé (si on estime)
5. approvisionner avec de nouvelles stories
6. purger le backlog



# Les 5 Bacs de Claude Aubry

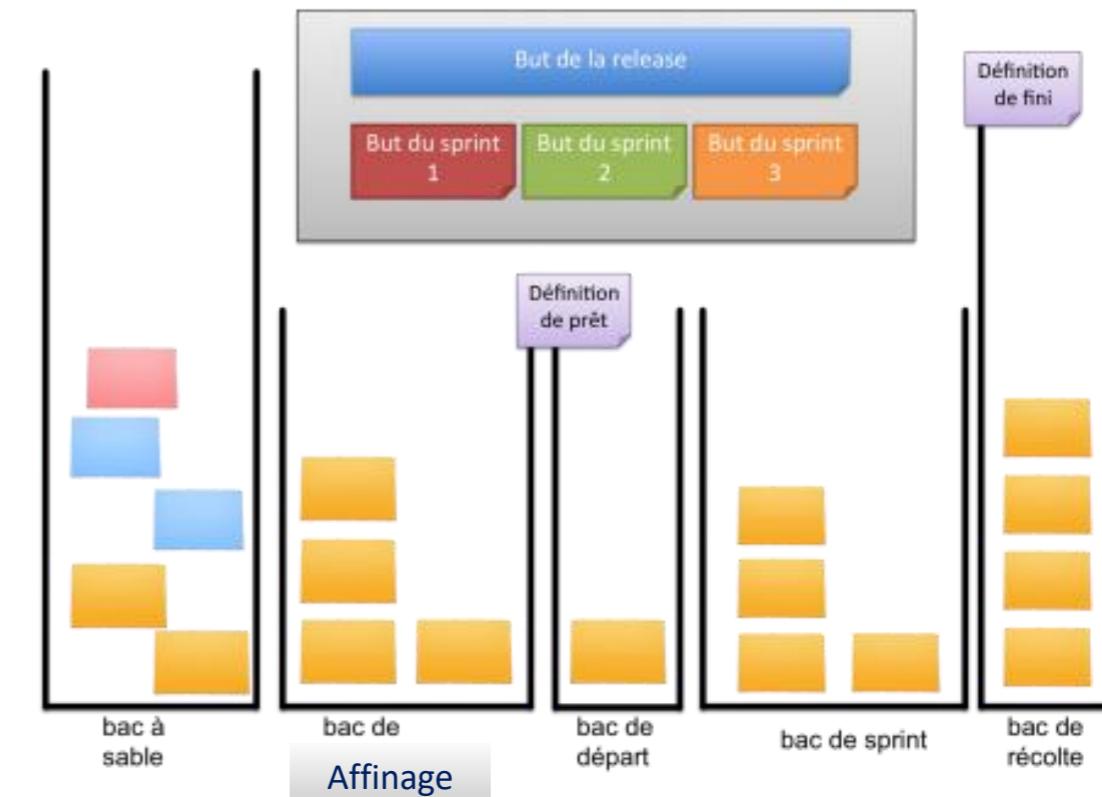
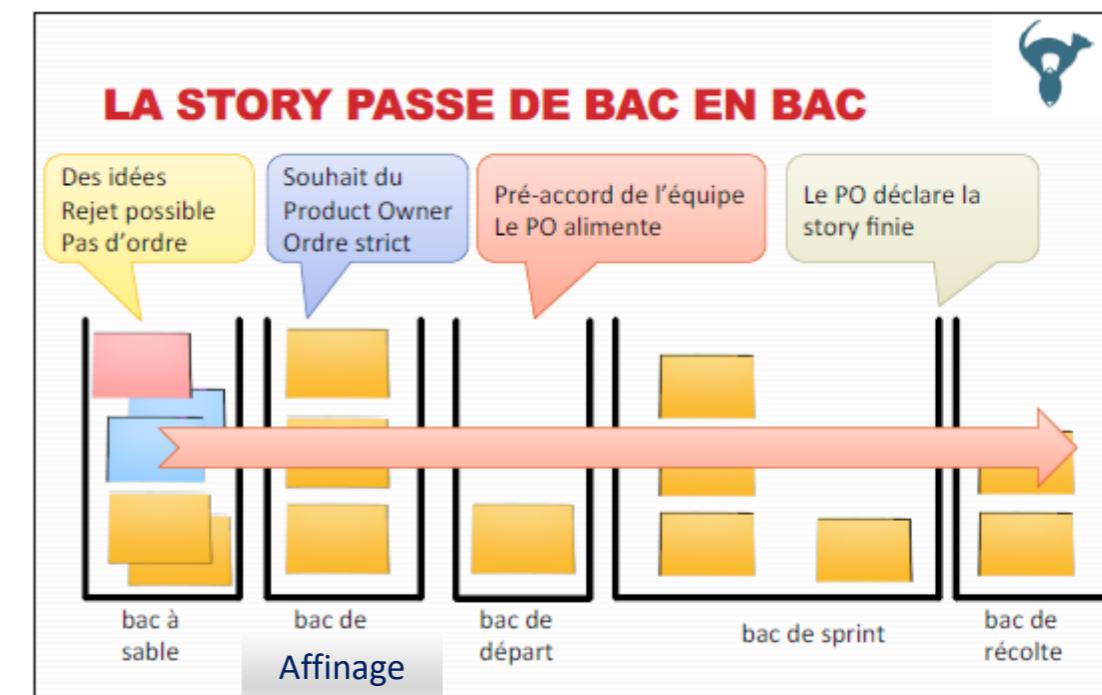


Extraits de « Les nouveaux outils du PO » :

<http://www.aubryconseil.com/post/Ma-presentation-au-ScrumDay-2014>

où les 5 bacs sont détaillés...

et de <http://www.aubryconseil.com/pages/Jeu-des-bacs-facon-puzzle> (un jeu à tester...)



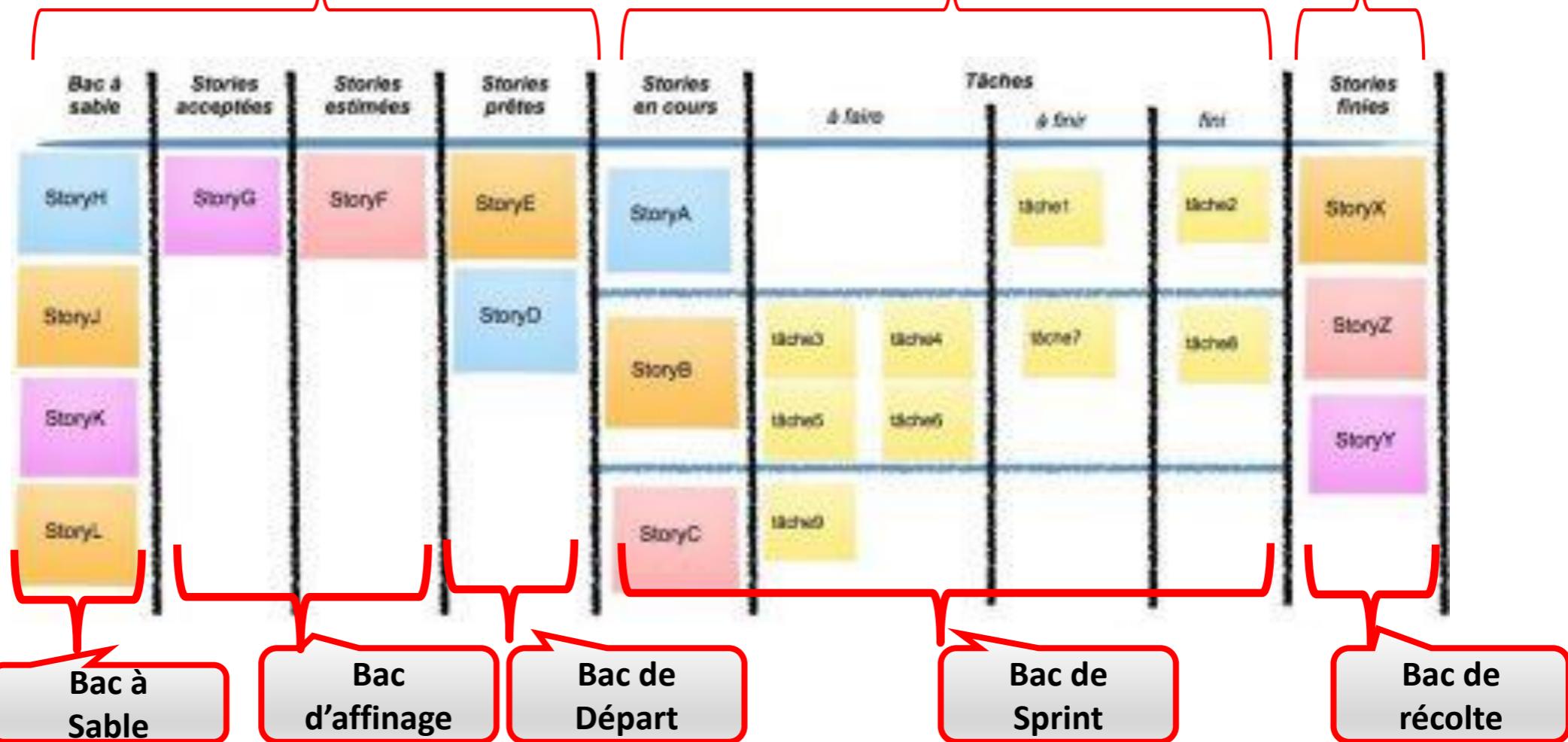
# Possibilité de kanbaniser le task board de Scrum

Pour information

Ajout de 4 colonnes pour suivre la **vie de la story avant le sprint**

Task board « classique » pendant le sprint

Ajout d'une colonne pour **déplacer les stories finies** (quand toutes les tâches associées sont finies)



Extrait : <http://www.aubryconseil.com/post/Tableau-a-2-niveaux>

# Conclusion

# Attention les User Stories ne sont pas des Use Case

Sur son blog, Claude Aubry répond à l'article précédent par un article intitulé **User stories et use-cases** (<http://www.aubryconseil.com/post/2007/04/09/203-user-stories-et-use-cases>) où il insiste sur le fait que :

...les histoires d'utilisateur, qui viennent de XP, reposent sur l'idée qu'il n'est pas efficace de rédiger des spécifications détaillées, mais qu'il est préférable :

- ✓ de dialoguer pour arriver au détail
  - ✓ de rédiger (ou d'écrire) des tests fonctionnels et de les passer pour valider l'histoire.
- Ces tests remplacent la spécification détaillée.



La définition d'une histoire d'utilisateur : **Un petit morceau fonctionnel qui a de la valeur pour le "métier" et qui peut être fourni en une itération** montre aussi leur autre facette, que n'ont pas les cas d'utilisation : la gestion de projet par l'intermédiaire du backlog de produit.

Les User Stories et les Use Case sont donc deux outils permettant de capturer de manière différente les besoins des utilisateurs. Pour souligner leur différence, Jean-Claude Grosjean a proposé sur son blog un autre article sur le sujet ([User Story vs Use Case : Soyez Agile !](#)) dans lequel il propose entre autre le tableau synthétique suivant qui reprend quelques différences entre la notion de Use Case et la notion de User Story.

USER STORY	USE CASE
Est une brève description d'une fonctionnalité telle que vue par l'utilisateur (Définition)	Représente une séquences d'actions qu'un système ou toute autre entité peut accomplir en interagissant avec les acteurs du système (Définition)
Format écrit <b>court</b> , laissant la part belle à la <b>discussion orale</b>	Format écrit <b>très riche en informations</b> (pré conditions, Evénement déclencheur, scénario principal, alternatives ...). Peu de place à l'oral
Est une <b>partie d'un Use Case</b> (le scénario principal <b>ou</b> une alternative)	Est la somme d'un scénario principal, et de diverses alternatives (variations, cas d'erreur ...)
Utilisée certes en tant que spécification mais <b>surtout pour l'estimation &amp; la planification</b>	Utilisé seulement en tant que spécification
Emergence rapide au travers d'ateliers de travail collaboratifs	<b>Long travail d'analyse et de formalisation</b>
<b>Grande lisibilité</b> du fait de sa simplicité	Manque de lisibilité même au sein d'un Template
<b>Mode ORAL et COLLABORATIF</b>	<b>Mode ECRIT ET DISTANT</b>
Très facile à maintenir (format fiche, court, indépendant)	Difficile à maintenir (Doc. Word de 150 pages)
Implémentée et testée <b>en une itération obligatoirement</b>	Implémenté et testé en une ou plusieurs itérations
Ecrive facilement par un Utilisateur ou un Client (accompagné dans sa démarche)	<b>Souvent rédigé par des User Proxies</b> (AMOA, Analyste ...), rarement par le client
<b>Contient des tests d'acceptation</b> (au dos de la carte) => implication des testeurs	Ne contient pas les cas de test qui en découlent => pas d'implication des testeurs
Difficile à lier les unes aux autres. Absence de vue globale	<b>Liaison et vision globale facilitée</b> : Sous Use Case, conditions préalables, Diagramme des cas utilisation
Associée historiquement à <b>eXtreme Programming et aux méthodes Agiles</b>	Associé historiquement au Processus Unifié
Un auteur de référence : <b>Mike Cohn</b>	Un auteur de référence : <b>Alistair Cockburn</b>

Pour plus d'informations, consultez les articles suivants :

<http://www.qualitystreet.fr/2009/02/16/user-story-vs-use-case-soyez-agile/>

<http://www.qualitystreet.fr/2007/03/30/user-stories-use-cases-les-differences/>

<http://www.aubryconseil.com/post/2007/04/09/203-user-stories-et-use-cases>

# Récapitulatif ...

## Agile Story Essentials

Use cards as the tokens for the conversations you'll use to plan, design, describe, construct, and validate your product

### Stories are for telling

In the late 1990's Kent Beck had a simple idea to solve one of the biggest challenges in software development: communicating the details of what to build. By simply getting together and "telling our stories" we could build shared understanding in the minds of everyone involved.

In the conversation we'd focus not only on what to build, but who would use the software and why. Our goal is to identify the most valuable thing we could most economically build.

*Stories get their name from how they're used and not how they're written*



Kent Beck, author of Extreme Programming Explained

What I was thinking of was the way users sometimes tell stories about the cool new things the software they use does:  
"I type in the zip code and it automatically fills in the city and state without me having to touch a button!"

I think that was the example that triggered the idea. If you can tell stories about what the user does and generate energy and interest and a vision in your listener's mind, then why not tell stories before the software does it?



For every story in your backlog, put in three cards. The first is what you want; the second one is there to remind you to fix the first one. The third is to remind you to fix it again. You've got to iterate or you're not doing it right.

### What's on the card

Use story cards, or items in backlog the way you might cards in a library card catalog. Write just enough information on them to help you find the rest of the details when you need to. Use the card or list item to organize stories, prioritize, and plan.

On a typical card you'll find:

**Short title** One that's easy to read in backlog and easy say in standup meetings  
If you catch yourself referring to the story by its number, stop it

**Description** If the title isn't enough, write a description. Try to include who, what, and why. The template could be handy here.

**Meta-Information** • Estimated development time  
• Estimated value  
• Dependencies  
• Status

### "Talk & Doc"

You'll have many discussions around stories with team members in a variety of roles. Draw pictures and record details as you do.

Bring models like workflow models, use cases, UI designs, or anything else that helps you explain the story. But, be prepared to modify it during the conversation.

Draw on whiteboards, model with post-it notes, or record on flipchart paper during your discussions.  
Keep models from your discussions as mementos to help you remember the details discussed.



©2013 Comakers LLC, www.comakewith.us, youshould@comakewith.us for more info



### Shared Understanding

When we all read the same document or hear the same discussion, we often imagine different things. It's describing our understanding with words and pictures, and then combining and refining our ideas that leads to shared understanding.

*Shared documents are not shared understanding*

### Card

Write your product ideas on cards, one per card.

### Conversation

Discuss your ideas with others. Let them ask lots of questions. Work together to come up with ideal solutions. The goal is to build shared understanding.



### Consequences

Now we've got working software to learn from. Those who originally asked for it and the builders evaluate. But, the software was likely for other users. You'll need to test the working software with them to see if it meets their needs. The goal is learning. And your ideas for improvement start the cycle again.



### Vacation Photos

### Confirmation

Bring models, personas, UI drawings or whatever you like into the conversation. Identify ideal solutions and draw new models. Work towards agreement on what to build. Record that agreement as a set of confirmation tests.



### Construction

Developers, testers, and others equipped with information from conversations and the shared understanding that comes with it build and test the software.



### Construction

Developers, testers, and others equipped with information from conversations and the shared understanding that comes with it build and test the software.



### Construction

Developers, testers, and others equipped with information from conversations and the shared understanding that comes with it build and test the software.



### Construction

Developers, testers, and others equipped with information from conversations and the shared understanding that comes with it build and test the software.



### Construction

Developers, testers, and others equipped with information from conversations and the shared understanding that comes with it build and test the software.



### Before you build, agree on what you're building

Before the team makes a commitment to build software described by a story, agree on acceptance criteria for the software. Record the answers to these questions:

- What will we test to confirm that this story is done?
- How will we demonstrate this software at a product review?



### Before you build, agree on what you're building

When writing story tests, use examples. See Goko Adzic's Specification by Example for valuable tips.



### Before you build, agree on what you're building

When writing story tests, use examples. See Goko Adzic's Specification by Example for valuable tips.

## Stories: Concept to Delivery

Progressively split and refine stories as you move them from vague idea through to working software

### Opportunities

Create an opportunity backlog from product ideas, and customer, user, and stakeholder requests.



### Opportunity Assessment

Before spending time going into details on any idea, discuss who the product, feature, or improvement is for, what benefit it brings by building it, and how much it could cost if it's similar to other solutions we've built. Use the results of this conversation to prioritize opportunities, and to make go/no-go decisions.

1. What problems are we solving, and for who?
2. What solutions will customers and users value?
3. What are usable solutions?
4. What's feasible to build given the time and tools we have?

### When splitting stories, think cake

Use each story to describe a piece of software you can "taste." That is, once you've built it, you should be able to learn something from having done so. Whole features may have value to customers and users. But, it often takes a few stories to add up to a whole feature.

The steps for making software are development tasks.

Demonstrable, testable software is the result of those tasks. If the software doesn't have user interface, you'll need to find another way to show that it works.

### Stories

Stories describe something to deliver and evaluate



Decompose stories into smaller deliverable stories

Smaller stories often have similar recipes, just less of any one ingredient. For example all stories will have some testing, smaller stories should take less time than larger stories.

### Delivery Tasks

Delivery tasks give the "recipe" – describe the work someone needs to do to create the story

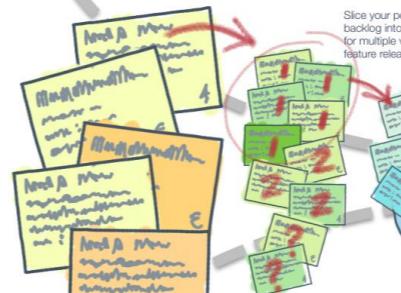


Decompose stories into smaller deliverable stories

Smaller stories often have similar recipes, just less of any one ingredient. For example all stories will have some testing, smaller stories should take less time than larger stories.

### Discovery

Use discovery to elaborate, design, and validate product ideas. Your goal is to identify the smallest viable product you can. Discovery work results in a product backlog.



### Product Team Planning

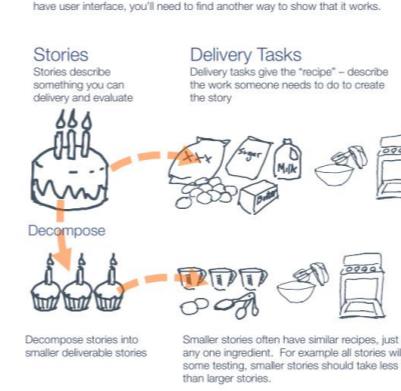
The product team meets routinely to discuss release progress, select stories for upcoming sprints/iterations, and plan the work needed to get stories ready for the delivery team.

- Story Workshop
- Product team members meet with delivery team member regularly to work through story details and agree on acceptance criteria
- Some call these workshops backlog refinement or backlog grooming meetings. But they're really the story conversations we need to have

### Work like da Vinci to finish on time

When managing a release budget, split larger stories into "opening game," "mid game" and "ending game" stories.

To get the "big picture" as soon as possible. Early versions that are fully formed but immature allow early functional and performance testing. They allow earlier validation that your concept is right.

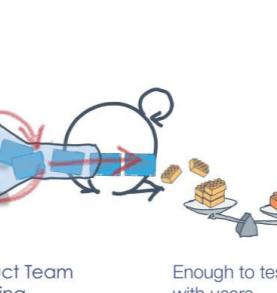


Decompose stories into smaller deliverable stories

Smaller stories often have similar recipes, just less of any one ingredient. For example all stories will have some testing, smaller stories should take less time than larger stories.

### Delivery

During delivery you'll focus on designing, decomposing, and describing backlog items.



### Validation

Review finished software with the team and stakeholders. Validate product parts with customers and users.

- Product
- Discovery
- Delivery
- Validation

### Release

After your software is released, continue to measure the product's performance relative to its target outcomes. The most valuable opportunities come after seeing the product in use.



### Release Strategy

During Discovery, try using a story map to slice a while product or feature into a series of viable releases.



Decompose stories into smaller deliverable stories

Smaller stories often have similar recipes, just less of any one ingredient. For example all stories will have some testing, smaller stories should take less time than larger stories.

Isabelle BLASQUEZ - 2016

# Annexes

# En savoir plus sur les User stories : la référence

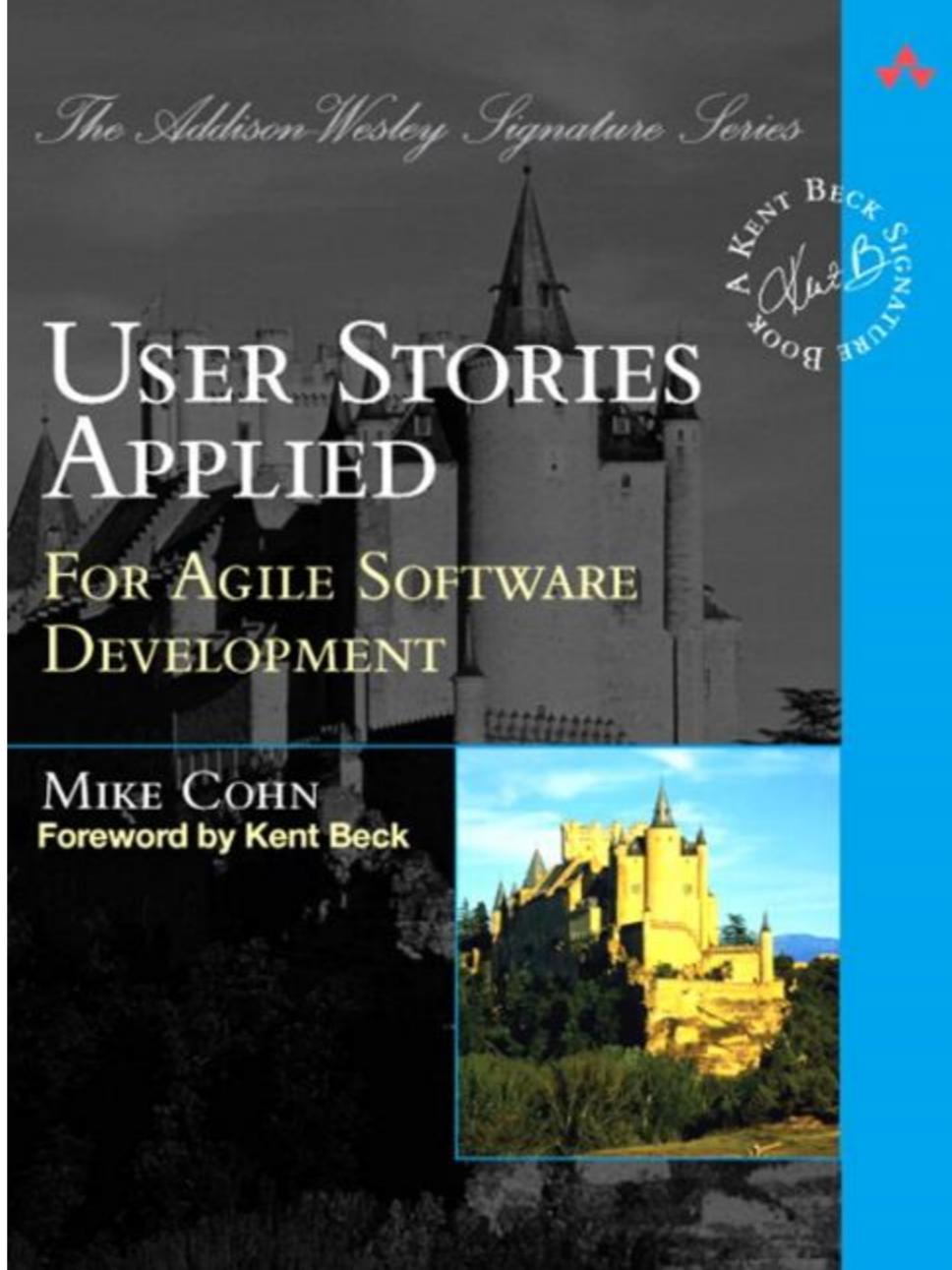
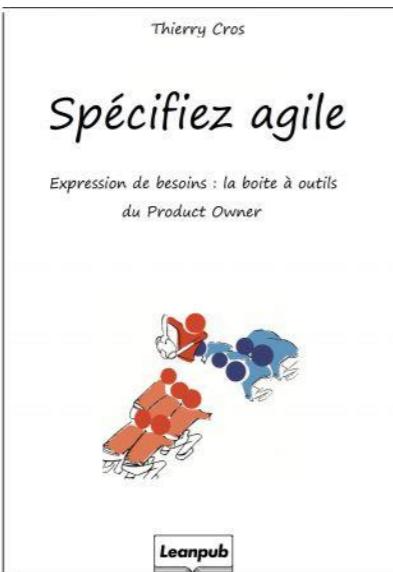


Table des matières disponible sur : <http://www.mountaingoatsoftware.com/books/user-stories-applied>

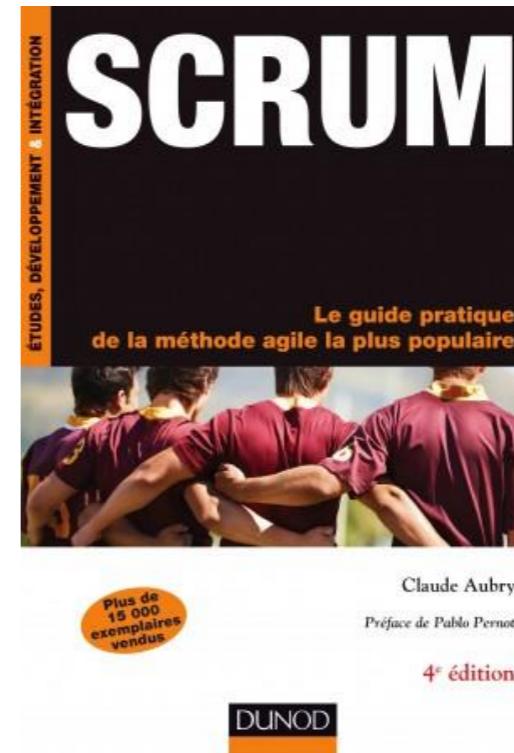
# En savoir plus sur les User stories : quelques lectures



<https://leanpub.com/agile-expression-de-besoins>



<https://leanpub.com/50quickideas>



<http://www.dunod.com/informatique-multimedia/developpement/methodes-modelisation-uml/ouvrages-professionnels/scrum>

# Intérêts de la décomposition ...

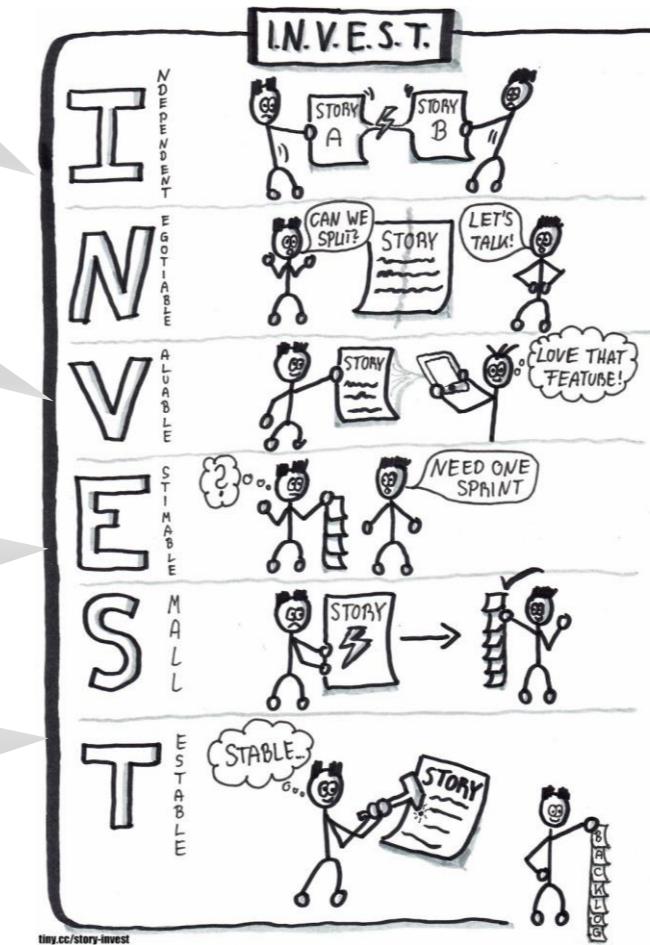
Identification plus facile des conditions d'acceptation

Le nombre de story peut servir de mesure de vélocité

Plus de flexibilité dans la planification

Finies plus vite et feedback plus rapide ...

Décomposer  
la story  
**(splitting)**



# Progressive refinement of user stories in the Product Backlog

*Progressive refinement by adding conditions of satisfaction helps team members by telling them the product owner's expectations for that feature.*

✓ **Split epics into smaller features**

✓ **Split epics into user stories**

✓ **Break large epics into smaller epics**

✓ **Add conditions of satisfaction**

✓ **Don't forget to talk**

# Atelier d'écriture des stories (sprint 0)

## Story-writing workshops

- Includes developers, users, customer, others
- Brainstorm to generate stories
- Goal is to write as many stories as possible
  - Some will be “implementation ready”
  - Others will be “epics”
- No prioritization at this point



© 2003–2008 Mountain Goat Software®

# Pertinence des critères d'acceptation

## Exemple de Critères d'acceptation

Planter un arbre

En tant que jardinier du dimanche  
Je veux planter un arbre fruitier  
Pour récolter des fruits l'été prochain

« How to Demo »

- Le pied de l'arbre est enterré
- Un engrais naturel est déposée au fond du trou
- La terre est tassée après la plantation
- La plantation est abondamment arrosée

## Critères d'acceptation à éviter

Planter un arbre

En tant que jardinier du dimanche  
Je veux planter un arbre fruitier  
Pour récolter des fruits l'été prochain

- Le trou fait 60 cm de diamètre et 80 cm de profondeur
- L'arbre est planté droit
- La pelouse est tondue autour de l'arbre

## Une User Story est orientée utilisateur

- La mise à disposition d'une User Story impacte l'utilisateur
- Raisonner « Valeur pour l'utilisateur » et non « Moyen de le faire »



Qu'est ce qui est important : Le moyen de faire le trou ou l'arbre ?

# Pertinence des critères d'acceptation



En tant qu'utilisateur

Je veux me connecter à Google

Afin d'accéder à tous mes services en ligne

L'utilisateur peut se connecter

Google trace la connexion de l'utilisateur

L'utilisateur peut accéder à tous ses services

La barre de menu google présente les services disponibles

Google présente la liste des nouvelles fonctionnalités disponibles

✓ Un critère d'acceptance doit être :

- Une **vision utilisateur**
- Ne **pas proposer de solution**
- Ne **pas être interne à la fonction**

✓ Un critère d'acceptance doit relevé de la story

*Ne Relève pas de la story*

# Suggestion de guides de conversation pour faciliter le récit et les détails de la story

Histoires Utilisateur - "user stories"

Titre de l'histoire utilisateur

En tant que <rôle>,  
Je peux/veux <besoin>  
De façon à <bénéfice/valeur>

Given : une situation donnée  
And : un contexte  
When : quand j'actionne, execute, fait, etc.  
Then : alors j'obtiens tel résultat

Given : une situation donnée  
When : quand j'actionne, execute, fait, etc.  
And : que j'actionne aussi  
Then : alors j'obtiens tel résultat

Pablo Pernot - 2013 - Creative Commons Attribution-ShareAlike 3.0 Unported License  
<http://creativecommons.org/licenses/by-sa/3.0/>

Maquettes

Titre

En tant que  
Je  
De façon à

étant donné  
quand  
alors ;  
étant donné  
quand  
alors ;  
étant donné  
quand  
alors ;

et  
et  
et  
et  
et  
et

Notes durant la conversation

Rappels  
 Indépendante  
 Négociable  
 Accès de la valeur  
 Estimable  
 Assez petite  
 Testable

Pablo Pernot - 2013 - Creative Commons Attribution-ShareAlike 3.0 Unported License  
<http://creativecommons.org/licenses/by-sa/3.0/>

**Pour information**

Ce sont juste  
**des guides pour aider la conversation**  
et ne doivent pas être considérés  
comme un n<sup>ième</sup> format de spécification !  
Lorsque c'est nécessaire, n'hésitez pas à  
ajouter tout élément complémentaire  
favorisant la bonne compréhension de la  
story, sa réalisation et ses tests

USER STORY CHECKLIST

[www.qualitystreet.fr](http://www.qualitystreet.fr)

Titre :	Créer une User Story
ID :	A01
Scénario :	En tant que < Rôle >, je peux < But > pour < Justification Business >
Estimation :	n Points
Priorité :	Haut

Conditions de satisfaction :

- Ex : « La livraison gratuite est offerte aux clients Français qui passent une 1ere commande »

Exemple :

- Au format Texte : « Il y a 5 items sur la page. J'en prends 1 à 20 euros et le mets dans mon caddie. Je vais à la page suivante qui contient 5 items, j'en prends 1 à 10 euros. Je décide de commander : je vois les 2 items et le total de 30 euros »
- OU
- Au format Table de données

Règles Métier :

- Lorem ipsum
- Lorem ipsum

Actes sur fonctions existantes / doc / Architecture :

- Lorem ipsum
- Lorem ipsum
- Lorem ipsum
- 

Référence à Spécification / Maquette :

Oui. Lien vers spécification [Lorem ipsum](#)/[Lorem ipsum](#)/[Lorem ipsum](#)  
Oui. Lien vers maquette ou dessin [ENPJ](#)

Premiers cas de test (dans les grandes lignes) :

- A01 T01 Lorem ipsum
- A01 T01 Lorem ipsum

Disponible sur :

<http://www.qualitystreet.fr/2012/02/09/user-story-checklist-soyez-pret-soyez-efficace/>

Disponible sur :

<http://www.arevouagile.com/pdf/expressions-du-besoin-1-1.pdf>

Isabelle BLASQUEZ - 2016

# Références liées à l'étude de cas Peetic

✓ **Peetic** : Etude de cas initialement proposée Pablo Pernot ([@pablopernot](#))

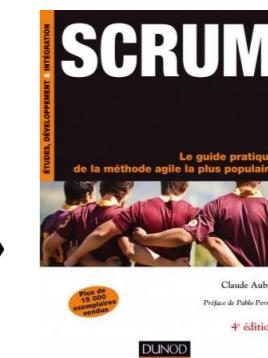
→ **Peetic, les différents épisodes**

- 2012 : [Episode 1](#)
- 2012 : [Conversations](#)
- 2014 : Un article de [l'idée au plan de livraison](#)
- 2015 : [Un peu de croquettes](#)

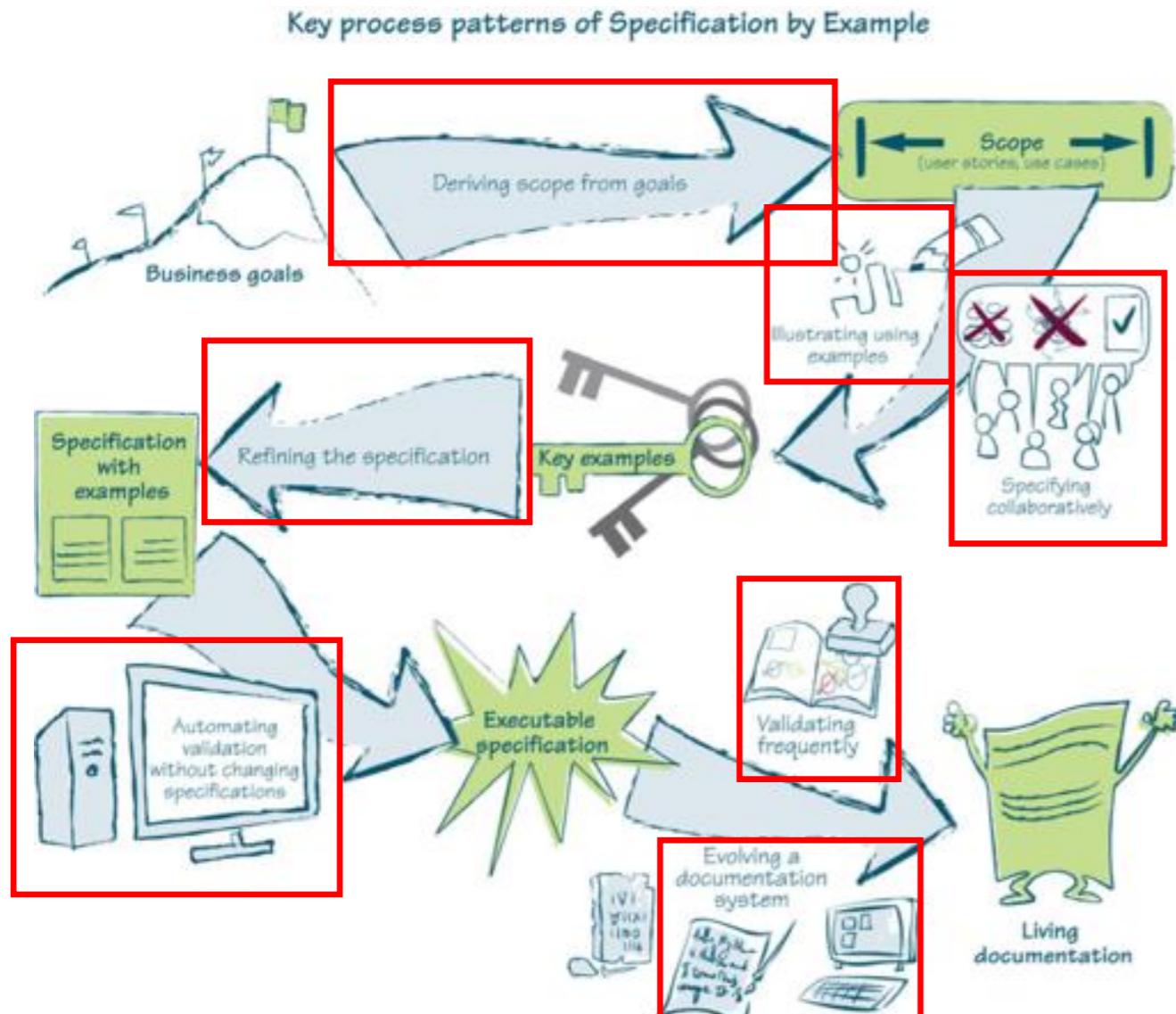


→ **Peetic, matériel en ligne** : <https://github.com/pablopernot/peetic>

→ **Peetic** est également le fil rouge utilisé par Claude Aubry ([@claudeaubry](#)) dans « Scrum : le guide de la méthode agile la plus populaire »



# Les 7 patterns de la spécification par l'exemple ...



## Deriving scope from goals

Travail autour de la vision

## Specifying collaboratively

Compréhension commune. La Collaboration permet aux équipes de produire des spécifications qui sont faciles à comprendre.

## Illustrating requirements using example

Exemples précis, complets, réalistes

## Refining specifications

Création d'un contexte concret et précis pour le développement et le test

## Automating validation without changing specification

Une **Spécification avec des exemples** automatisés (tests) qui est compréhensible et accessible par tous les membres de l'équipe devient une **spécification exécutable**

## Validating frequently

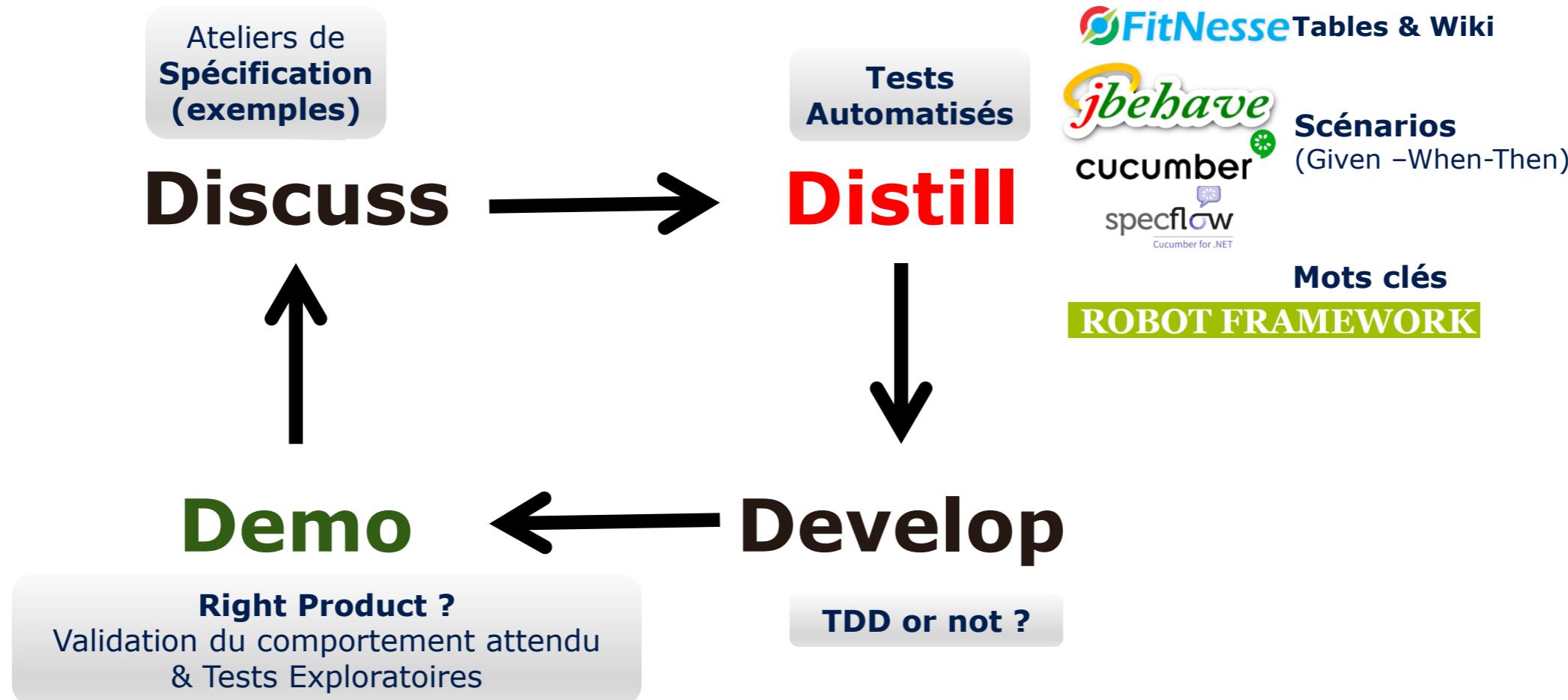
## Evolving a documentation system

Proposer une **documentation vivante** : facile à comprendre, cohérente et organisée

Extrait : [https://en.wikipedia.org/wiki/Specification\\_by\\_example](https://en.wikipedia.org/wiki/Specification_by_example)

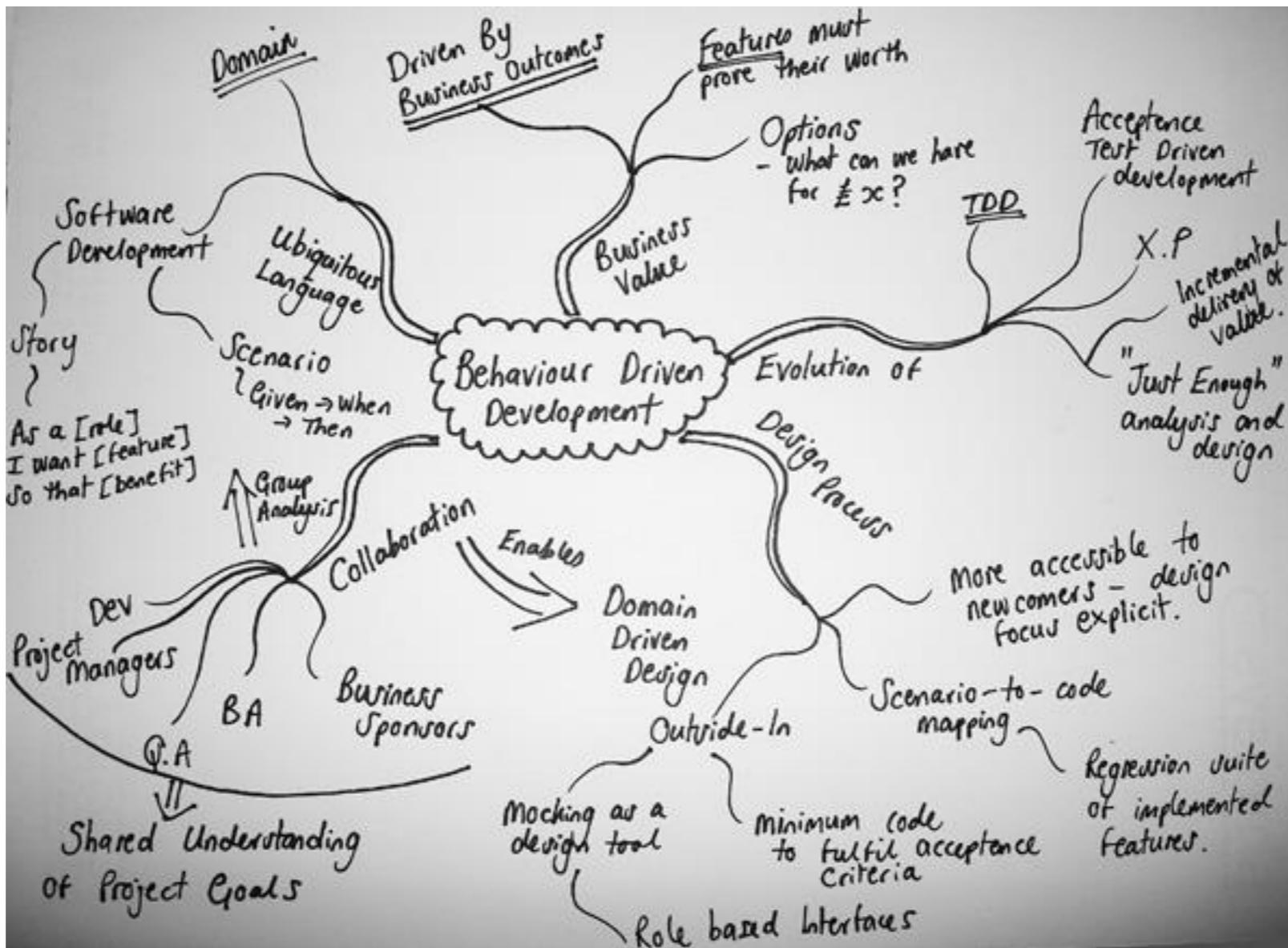
# Développement dirigé par les tests d'acceptation

## ATDD (Acceptance Test Driven Development)



A lire : <http://testobsessed.com/wp-content/uploads/2011/04/atddexample.pdf>

# Poursuivre votre réflexion sur le BDD



Extrait de : <http://ryangreenhall.blogspot.fr/2008/10/mind-mapping-behaviour-driven.html>

A lire également : Interviews Croisés : Couverture de Code, TDD et BDD (<http://www.infoq.com/fr/articles/virtual-panel-tdd-bdd>)

Isabelle BLASQUEZ - 2016