



# A la découverte de l'Example Mapping !

*LimouziCodev – Coding/PO Dojo #2*

*Juin 2023*

*Isabelle BLASQUEZ  
@iblasquez*

Cet atelier s'inspire fortement  
de l'atelier joué initialement  
à Agile Tour Toulouse 2023  
(<https://tour.agiletoulouse.fr/>)



# Quarto : Des specifications !





# Quarto : Démo !

## Immersion dans le métier !



# Exemple(s) Let's go !

## Story

En tant que joueur

Je souhaite **déetecter un quarto**  
afin de gagner la partie !



**Exemple(s)  
déetecter  
un quarto**



**Exemple(s)  
ne pas  
déetecter un  
quarto**



Un exemple par carte  
Une question par carte

**Question**  
(si elle se présente)

**Il n'y a pas de questions stupides.**  
Amusez-vous et prenez vraiment le temps d'**explorer** le problème.



# Mise en commun des exemples



Question(s)  
en suspens

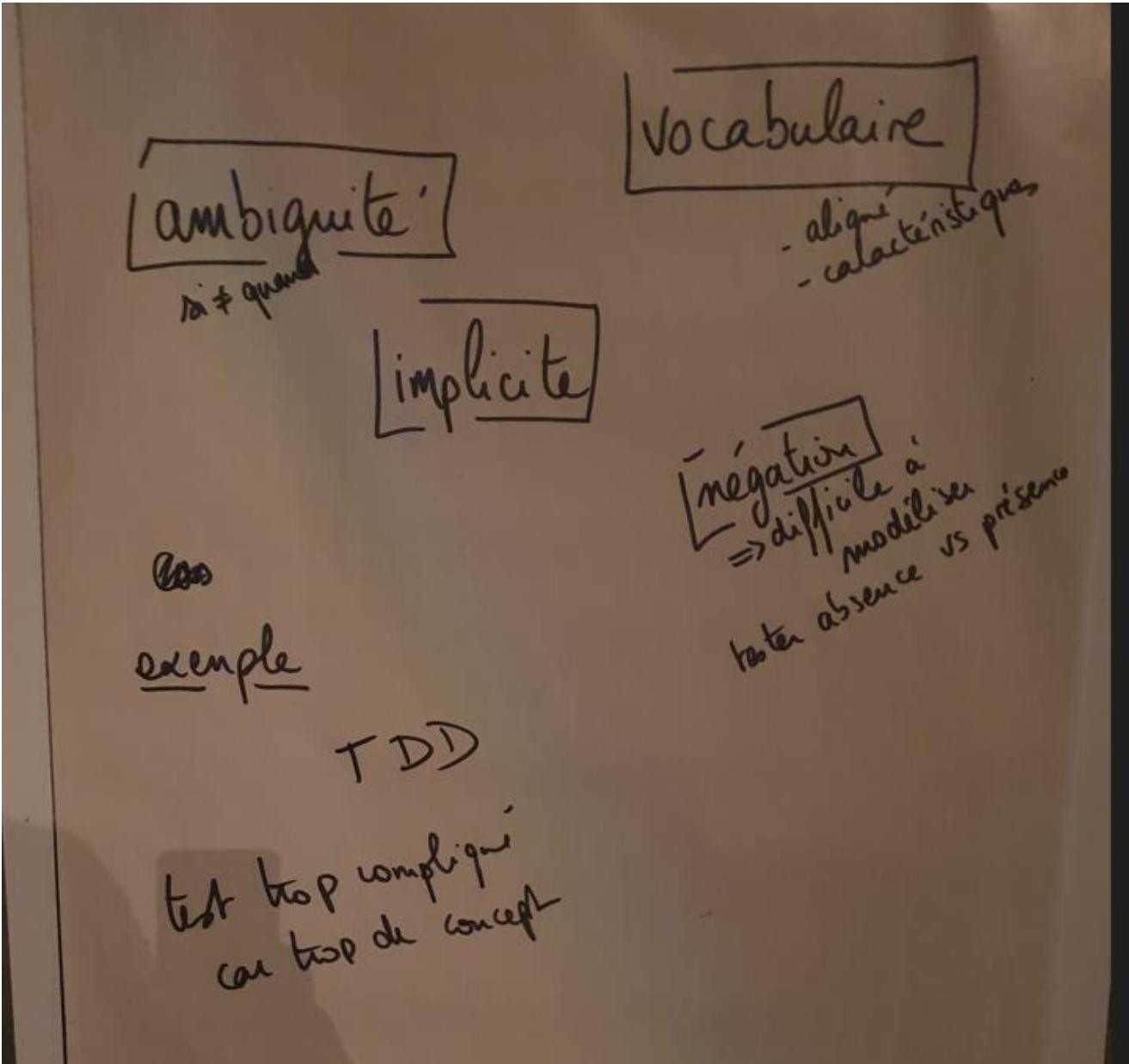
Combien avez-vous  
écrit d'exemples ?

Quelle est la forme  
des exemples  
écrits ?

A la lecture,  
les exemples sont-ils  
ambigus ? répétitifs ?

Le vocabulaire utilisé pour  
écrire les exemples est-il  
Compris de tous ?  
Non ambigu ?  
cohérent et uniforme ?

# Remarques sur la rédaction des exemples ...



# Rédiger des exemples ...

**La rédaction/illustration des exemples doit être *simple* et précise pour être rapidement compréhensible par tous et lever toute ambiguïté.**  
*(rester dans le métier, pas de référence aux technologies ou aux interfaces graphiques)*

Suivant l'utilisation que l'on veut faire de l'exemple Mapping :

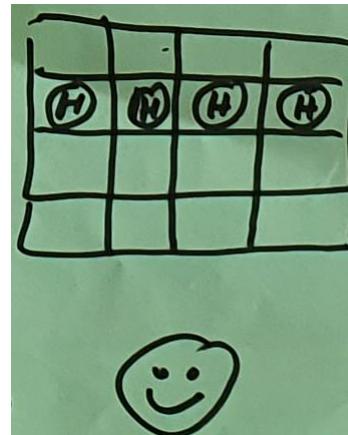
- établir une liste **d'exemples bruts**  
reprenant la convention de nommage des épisodes de Friends  
**(*The One who : celui dans lequel*)**

. CELUI DANS LEQUEL 4 PIECES HAUTES SONT ALIGNÉS SUR LA DEUXIÈME LIGNE

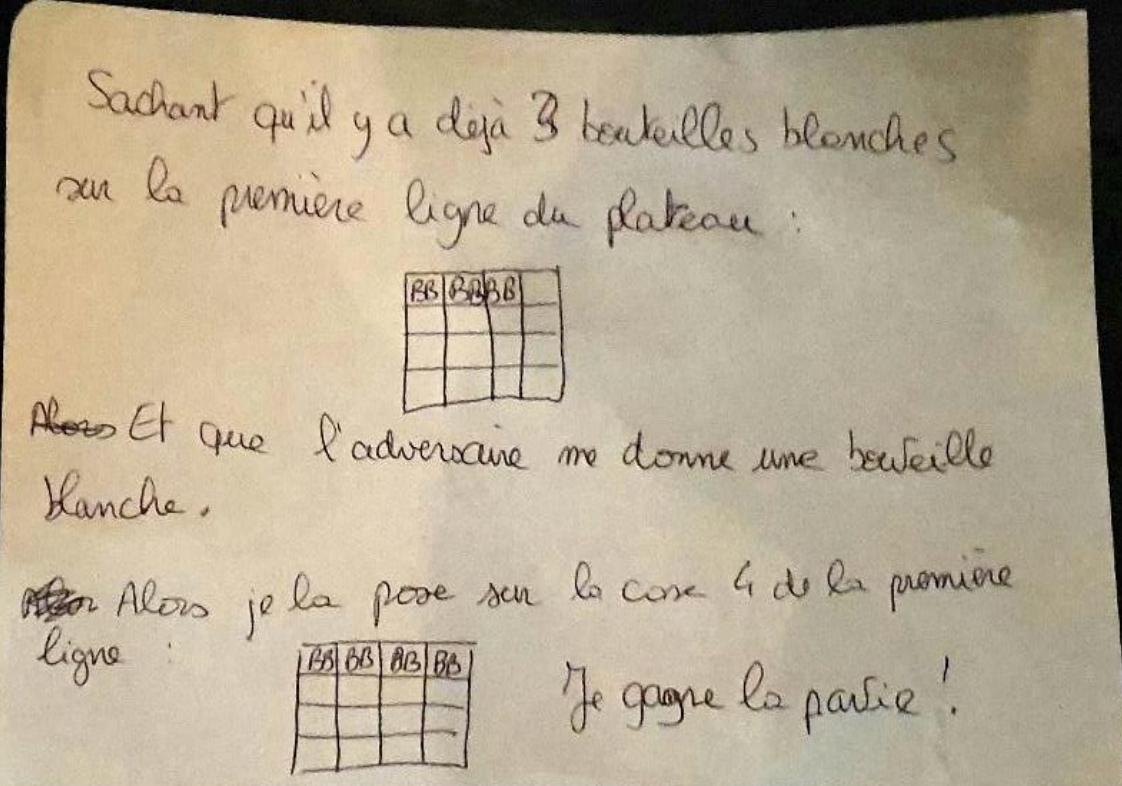
- scenarii à la Gherkin avec des valeurs (pattern AAA)

- 3 PIECES HAUTES DEJA PRÉSENTES SUR LES 3 PREMIERES CASES DE LA DEUXIÈME LIGNE
- ON POSE UNE PIÈCE HAUTE SUR LA 4ÈME CASE DE LA DEUXIÈME LIGNE
- QUARTO DETECTE !!!

... Parfois, lorsque l'incertitude monte, instinctivement vous pourriez avoir besoin de quelque chose de plus concret (structuré, précis) possibilité **d'ajouter des dessins** pour lever l'ambiguïté ;-)



# Zoom sur un exemple parmi d'autres (Quarto DIY)



## Anecdote :

- La consigne était :  
Exemple(s) pour gagner la partie ...
- Pour cette équipe,  
Seul exemple produit dans le temps imparti

... mais ce fut l'exemple choisi par le développeur pour implementation derrière

Mais au fait,  
pourquoi  
pouvez-vous  
dire “quarto!”  
dans ce cas ?

---



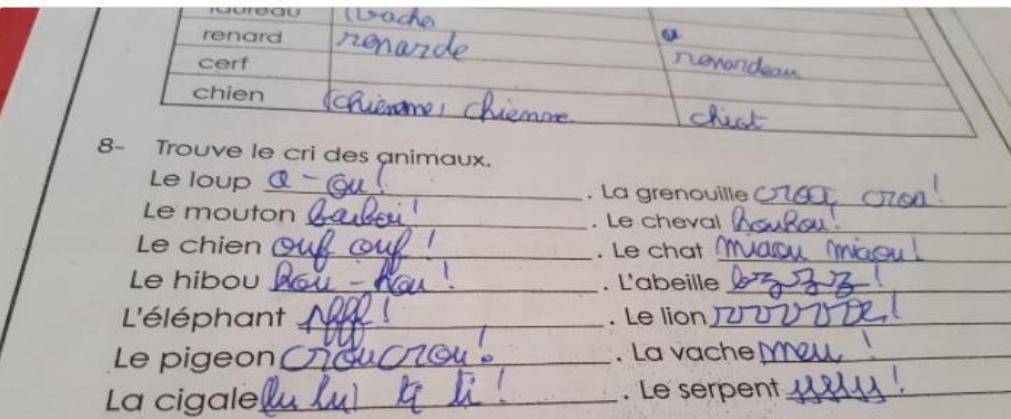
Nous avons besoin  
de règles ET d'exemples !!!!

utiliser des exemples  
Permet d'illustrer les règles



**Ninoche**  
@Prof\_Ninoche

Parfois quand t'es instit, tu imagines que ta consigne est claire et puis ... #Gilbert



**Nils Lesieur**  
@Nils\_Back

La spé semble claire et précise mais la réponse n'est pas celle attendue... =>  
**#SpecificationsByExample !!**

**3 Écris en chiffres les nombres suivants. ★★**

- a. dix-sept → 18
- b. onze → 12
- c. quatorze → 15
- d. quatre

**4 Écris en lettres les nombres suivants. ★★**

- a. 15 → quinze
- b. 12 → treize
- c. 19 → neuf
- d. 13 → quatorze
- e. 7 → huit
- f. 18 → dix-huit

Good job, kid.

[bit.ly/2r6qO4g](https://bit.ly/2r6qO4g)

2. 1 fire  
3 heat  
2 gas

4. 1 fer  
2 me  
3 nec

C. Write these words in alphabetical order.

1. take

A E K T

value

A E L U V

use

E S U

2. royal

A L O R Y

car les règles seules ne suffisent clairement pas !

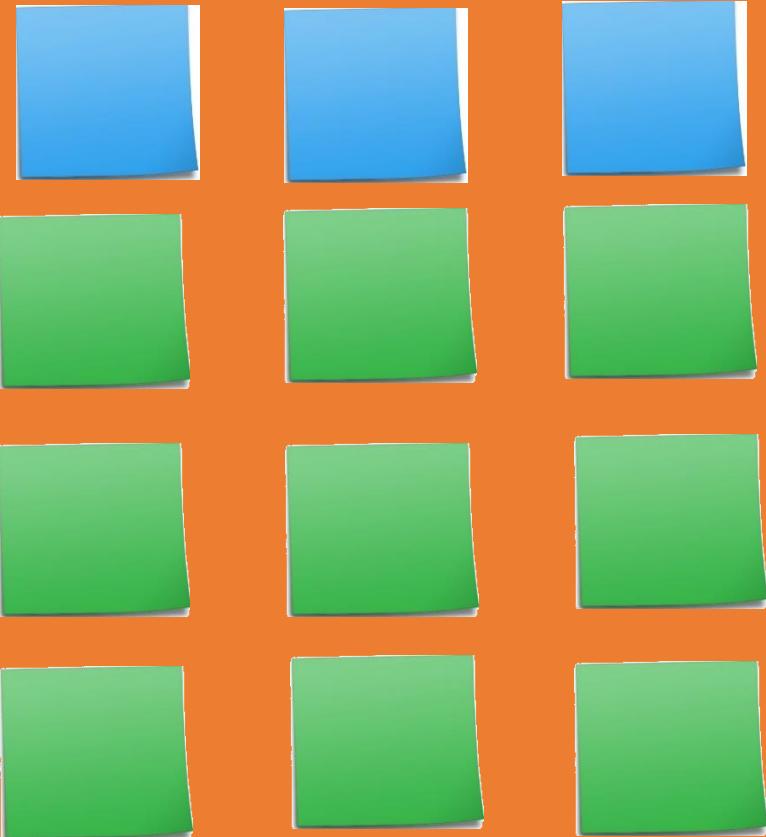
Ecriture de règle(s)  
pour illustrer  
vos exemples



Un exemple par carte  
Une question par carte (si nécessaire)



# Mise en commun des règles et des exemples



Question(s)  
en suspens



*1. Identifier la fonctionnalité*

## Story

*2. Identifier les règles*

Règle

Règle

Règle

*3. Lister tous les exemples*

Exemple

Exemple

Exemple

Exemple

Exemple

Exemple

Exemple

Exemple

Exemple

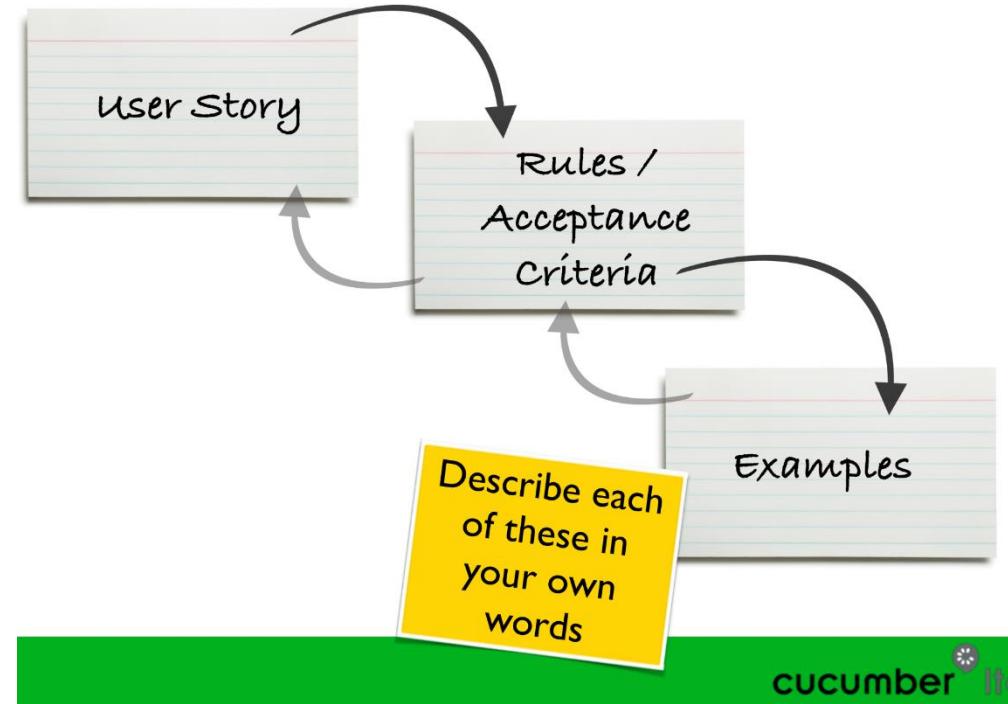
# Example Mapping : Récapitulatif

Question Question

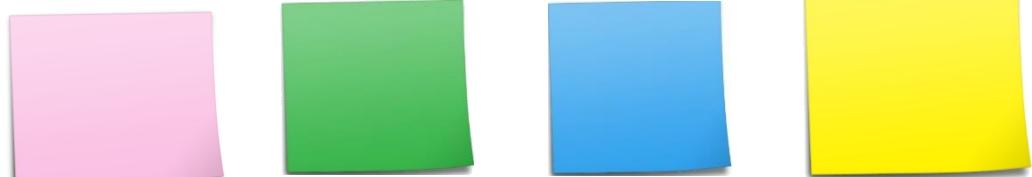
*A tout moment,  
noter des questions !*

OU  
cartographie des  
exemples

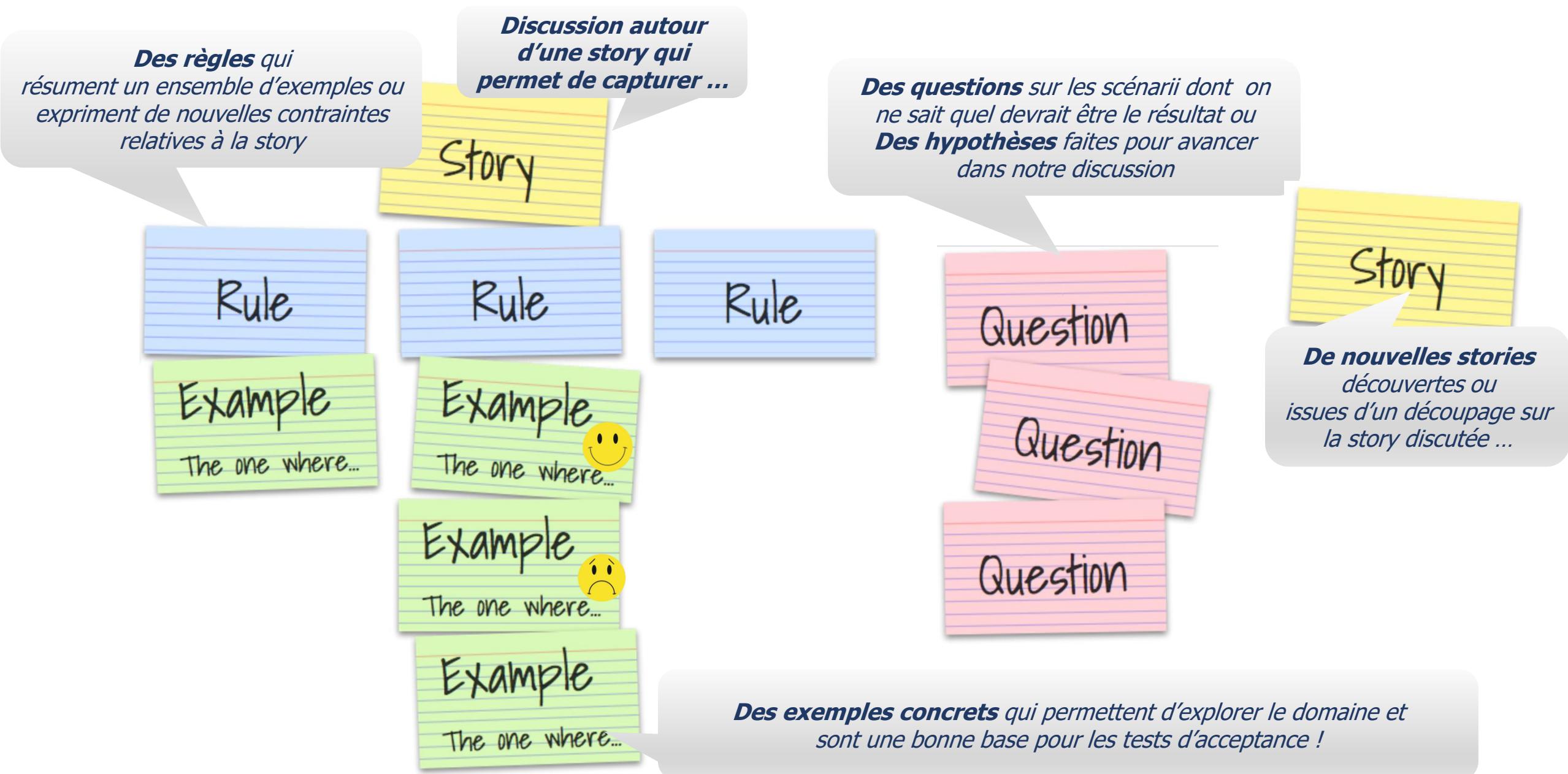
Les **exemples concrets** sont des moyens formidables pour nous permettre **d'explorer le domaine d'un problème**,



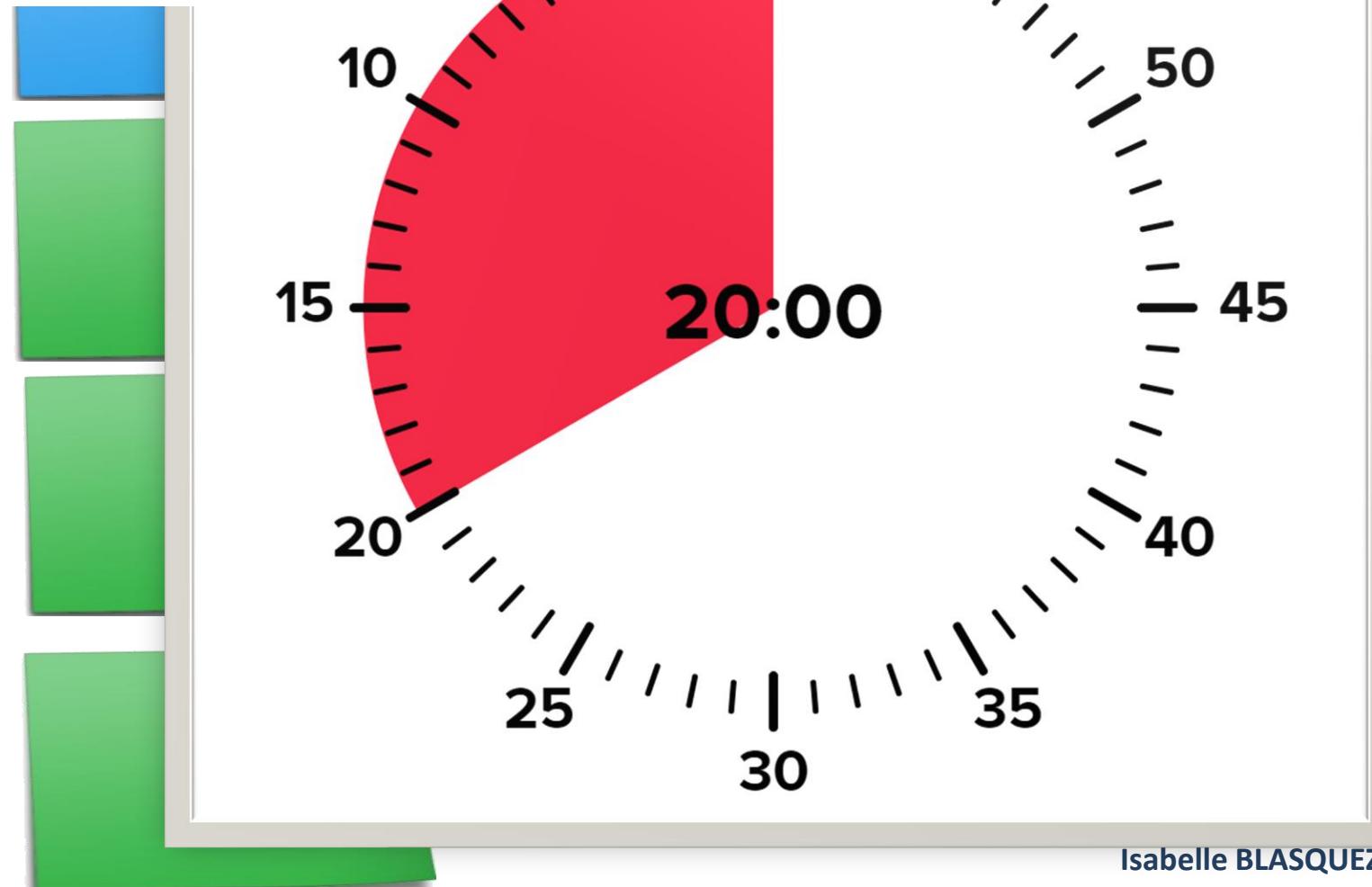
Au fur et à mesure que la discussion autour des exemples avance, d'autres choses émergent des conversations qui méritent d'être notées...



# Example Mapping : Mode Emploi !



# Si nécessaire, Timeboxer la discussion autour d'un même exemple mapping



# Example Mapping : Feedback !

Question

Trop de rouge sur la table indique que nous avons encore (beaucoup) à apprendre sur cette story ...  
Une nouvelle discussion est-elle nécessaire ?

Rule

Trop de bleu indique que cette story est sûrement trop *grosse* et trop *complexe*...  
Peut-être serait-il bon de la découper en nouvelles stories (jaune) ?



Example  
The one where...

Trop de vert (exemples) pour une même règle indique sûrement que la règle est trop *complexe*...  
D'autres règles ne pourraient-elles pas être identifiées ?

# VOTE pour savoir si on en sait assez sur la story ou s'il faudra encore l'affiner et en rediscuter ...



Chez Cucumber, après 25 minutes de conversation nous utilisons une technique de vote très rapide, celle du **vote avec le pouce**, pour **déterminer si la story est prête à aller en développement.**

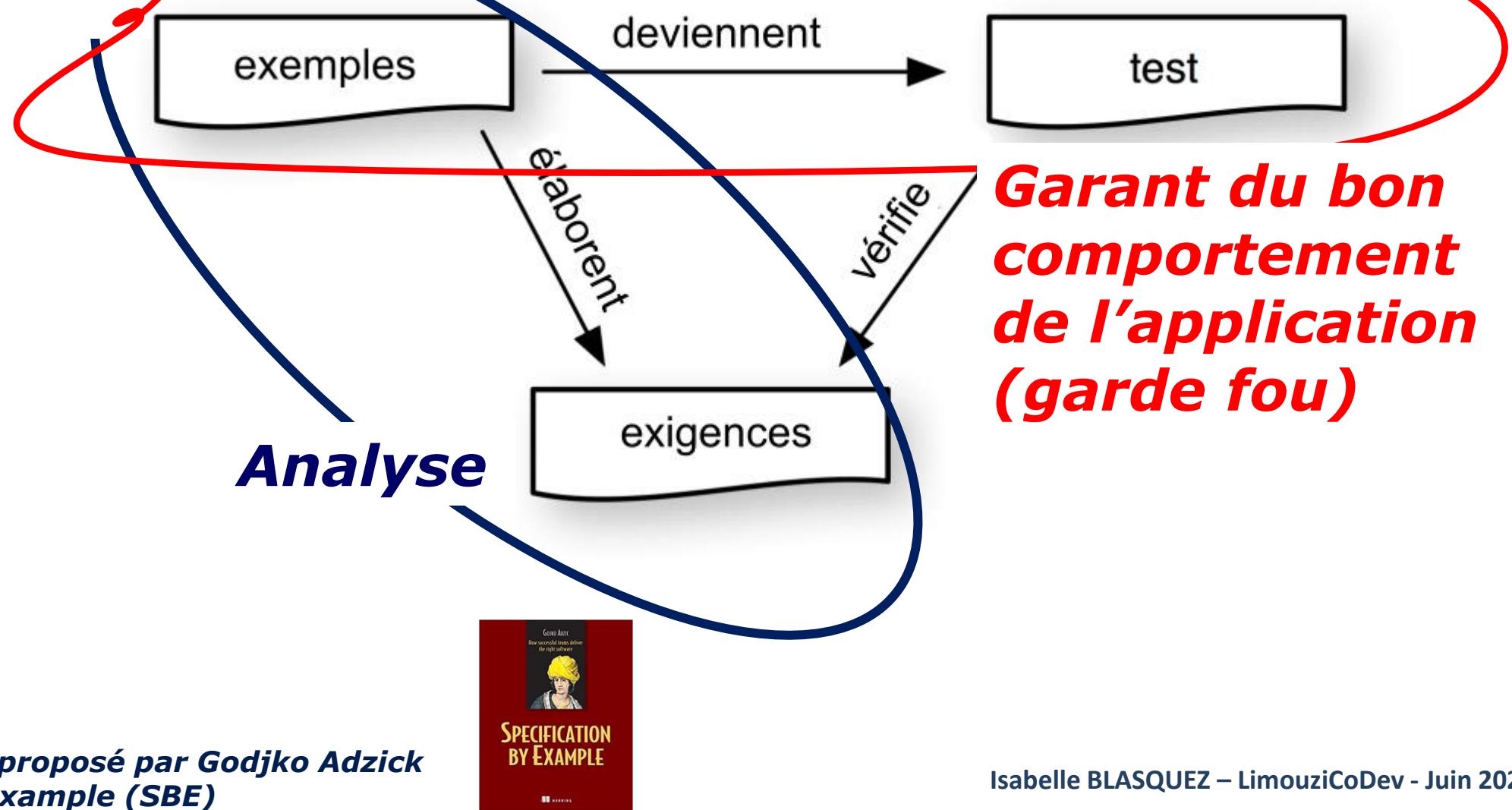


Même quelques questions persistent, vous pouvez estimer qu'elles sont suffisamment secondaires et que vous pourrez les résoudre lorsque vous y travaillerez.  
Laissons-le groupe décider.

# Et après ?

**Analyse**  
*(Elaboration des exemples)*

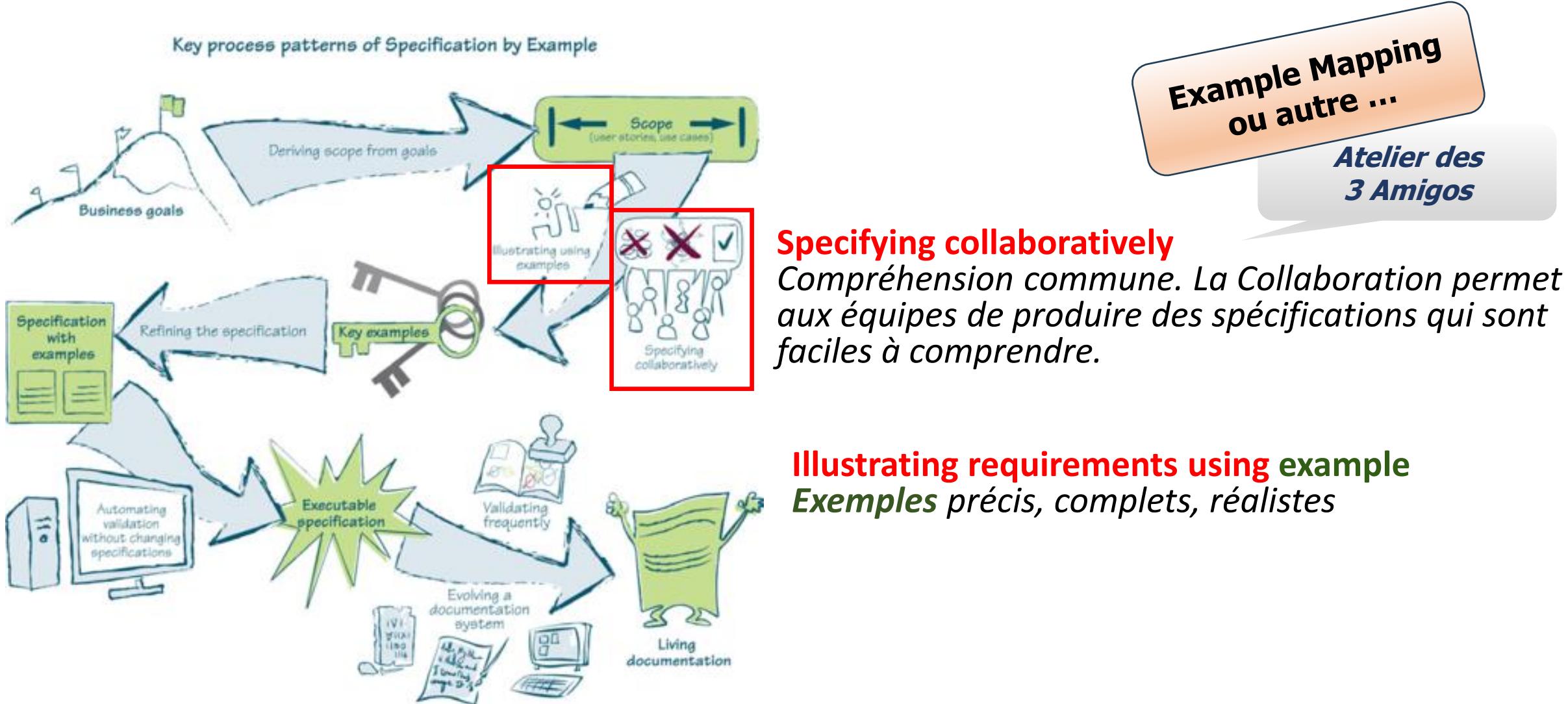
**Documentation**  
*(Re)lecture des exemples*





# L'Exemple Mapping dans la SBE : une implémentation possible des patterns

## Specifying collaboratively et Illustrating requirements using example



### Specifying collaboratively

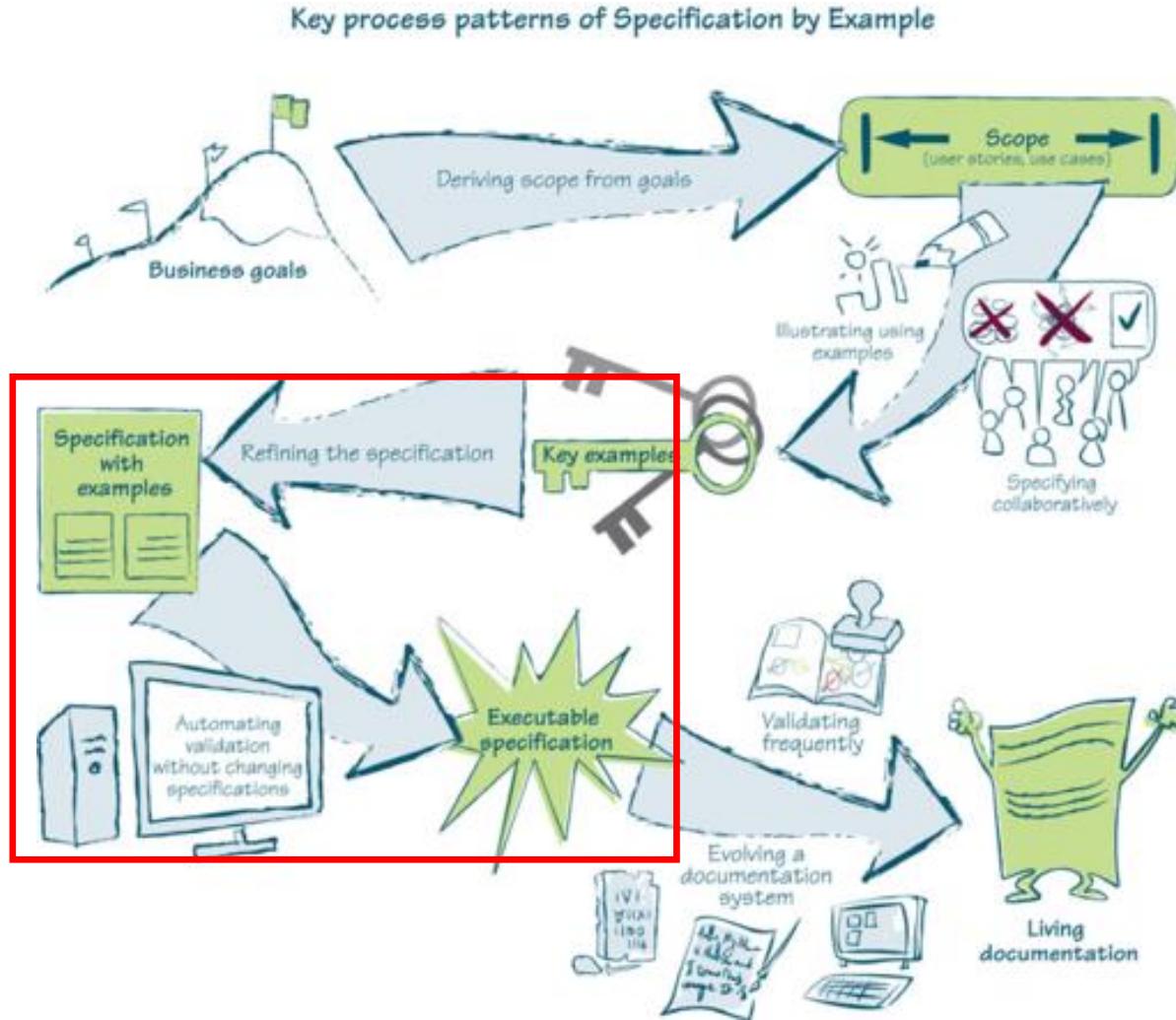
Compréhension commune. La Collaboration permet aux équipes de produire des spécifications qui sont faciles à comprendre.

### Illustrating requirements using example

*Exemples précis, complets, réalistes*



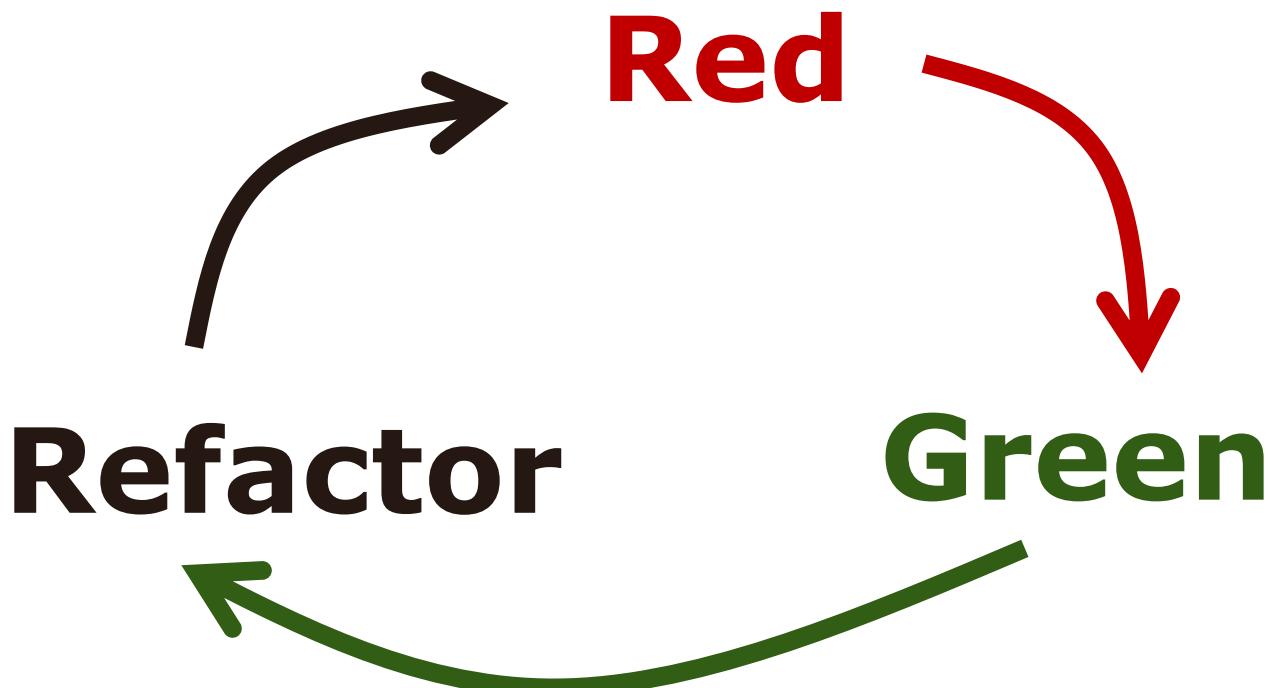
# L'Exemple Mapping : une aide pour obtenir des specs prêtes à coder



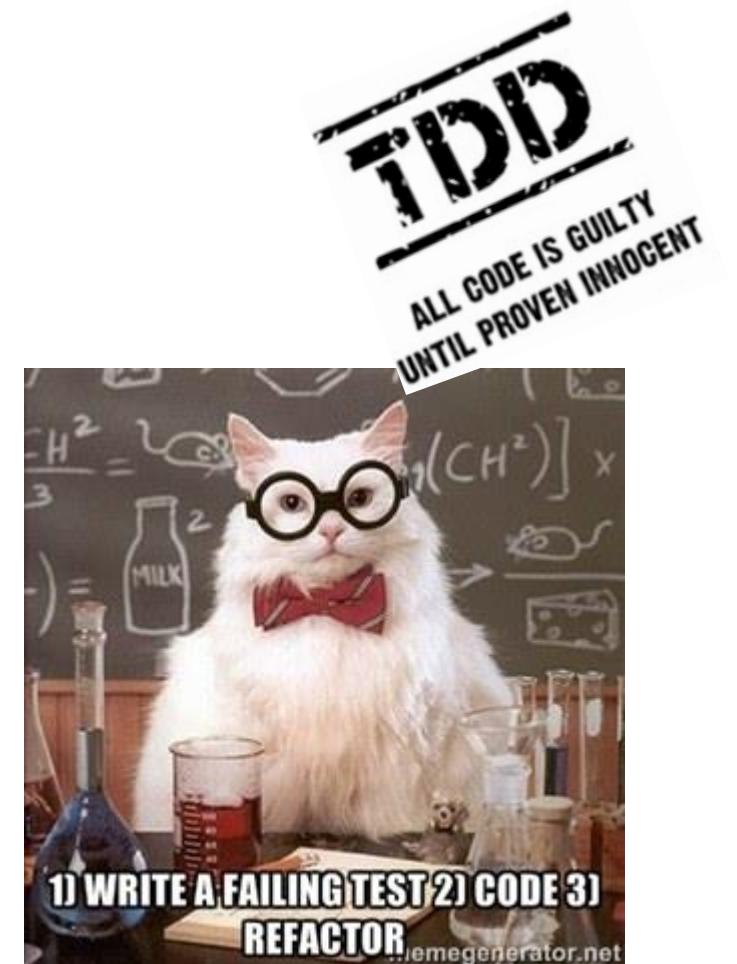
# Quarto : Des specifications prêtes à coder !



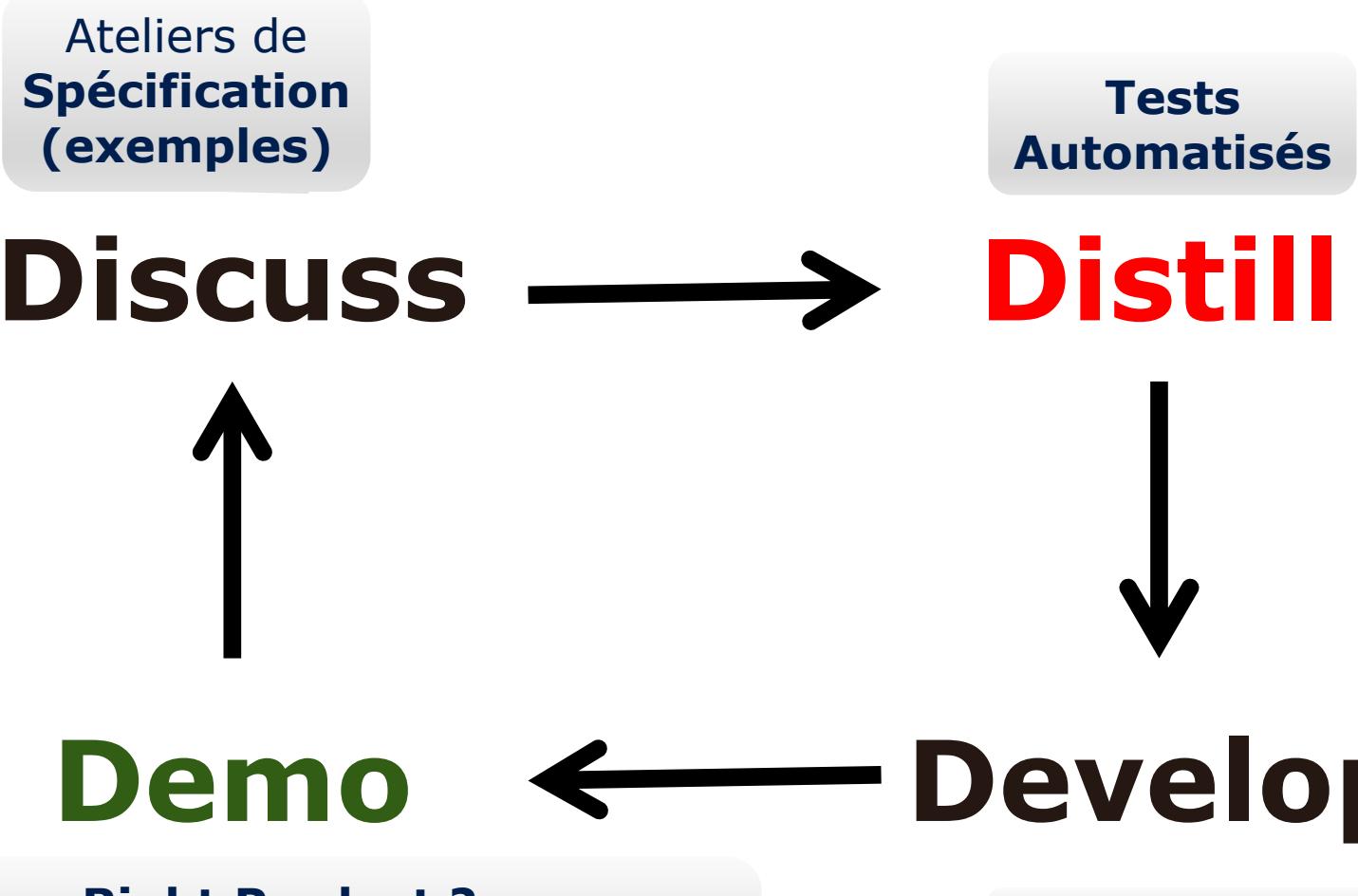
# Test Driven Développement



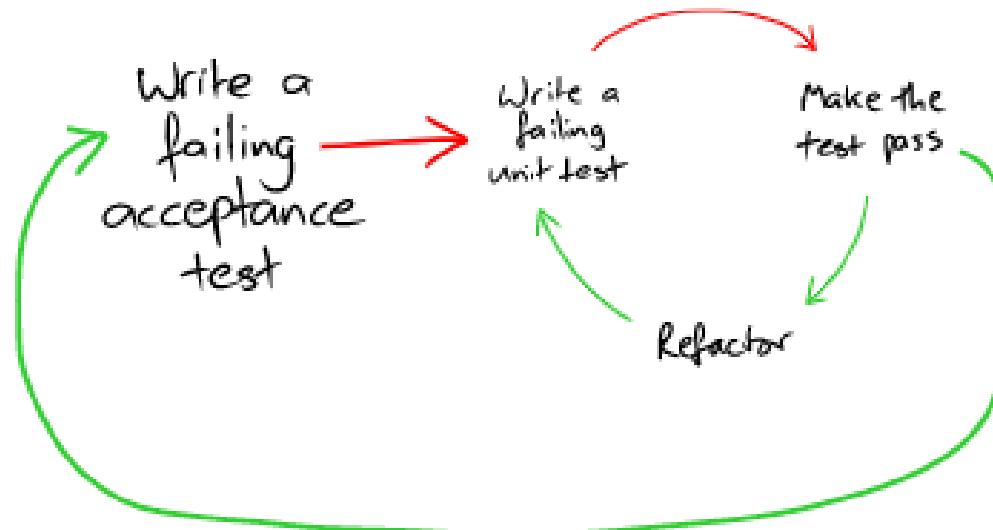
**Itérations « baby-step » rapides**  
*(de quelques secondes à quelques minutes)*



# Acceptance Test Driven Développement

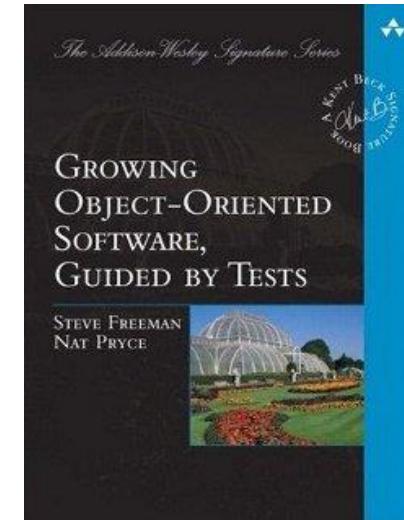


# Double Loop : Rappel



TDD with acceptance- and unit-test cycles

Le Double Loop TDD consiste à ajouter un test d'acceptance qui échoue (un test BDD généralement), puis ensuite implémenter la fonctionnalité ajoutée à travers une ou plusieurs boucles TDD



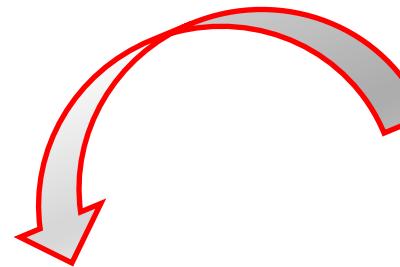
# Vers un premier test acceptance ...

→ On va repartir d'un **exemple** écrit lors de l'Example Mapping :

→ ... que l'on va écrire en Gherkin

- **Etant donné que** je joue au quarto  
**Et qu'** il y a une pièce claire sur la première case de la première ligne  
**Et qu'** il y a une pièce claire sur la deuxième case de la première ligne  
**Et qu'** il y a une pièce claire sur la troisième case de la première ligne
- **Quand** je pose une pièce claire sur la quatrième case de la première ligne

⇒ **Alors** un quarto est détecté



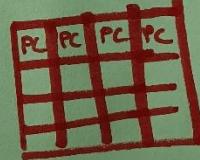
Sachant qu'il y a déjà 3 pièces claires sur la première ligne du plateau



PC = Pièce claire

Et que l'adversaire me donne une pièce claire que je pose sur la case 4 de la première ligne

Alors un quarto est détecté



# En savoir plus sur le Gherkin

[https://github.com/iblasquez/tuto\\_bdd\\_gherkin](https://github.com/iblasquez/tuto_bdd_gherkin)

## Le Gherkin, la(es) classe(s) !

Le langage **Gherkin** permet d'écrire des scénarios de test rédigés à l'aide de mots clés tels que **Given**, **When**, **Then** (**Etant donné**, **Quand**, **Alors**) dans un langage naturel lisible et compréhensible par tous (notamment par les gens côté métier). A ce titre, Gherkin est considéré comme un **DSL** (Domain Specific Language). Martin Fowler qualifie même ce genre de DSL de **BusinessReadableDSL**, un langage qui permet de décrire le comportement d'un système sans détailler comment ce comportement est implémenté.

Gherkin est, à la base, le langage utilisé par **Cucumber**, un framework de tests fonctionnels automatisés dont le site de référence est [cucumber.io](http://cucumber.io).

Pour découvrir et comprendre la grammaire du langage Gherkin, nous utiliserons des exemples extraits :

- du wiki de Cucumber : <https://github.com/cucumber/cucumber/wiki>
- de la section Gherkin du manuel Cucumber de référence : <https://cucumber.io/docs/reference#gherkin>
- et du livre **The Cucumber Book – Behaviour Driven Development for Testers and Developers** de Matt Wynne et Aslak Hellesoy

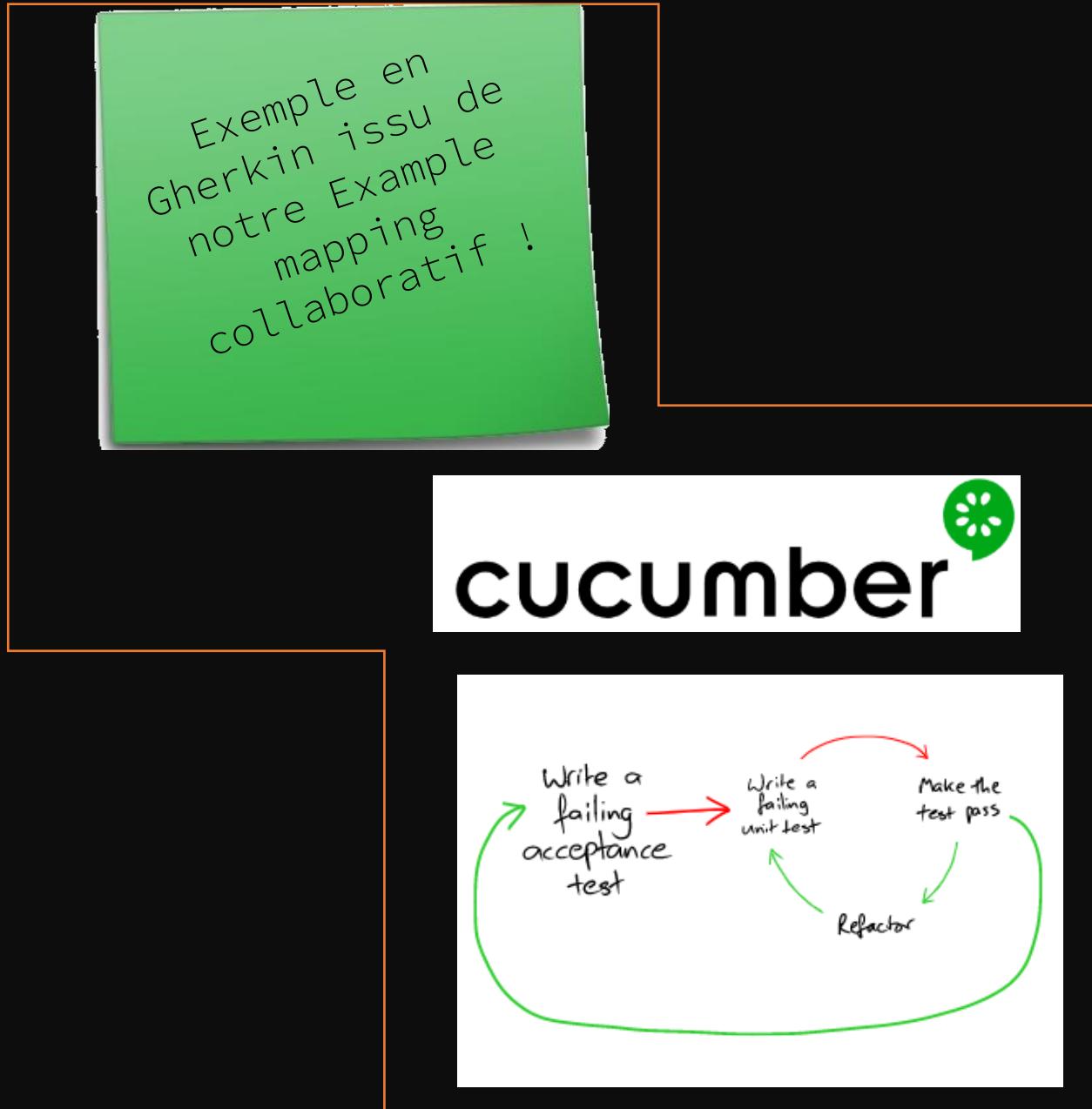


Grâce à ces différents exemples, nous procéderons pas à pas à la modélisation du langage Gherkin sous forme de diagramme de classes à partir des étapes suivantes :

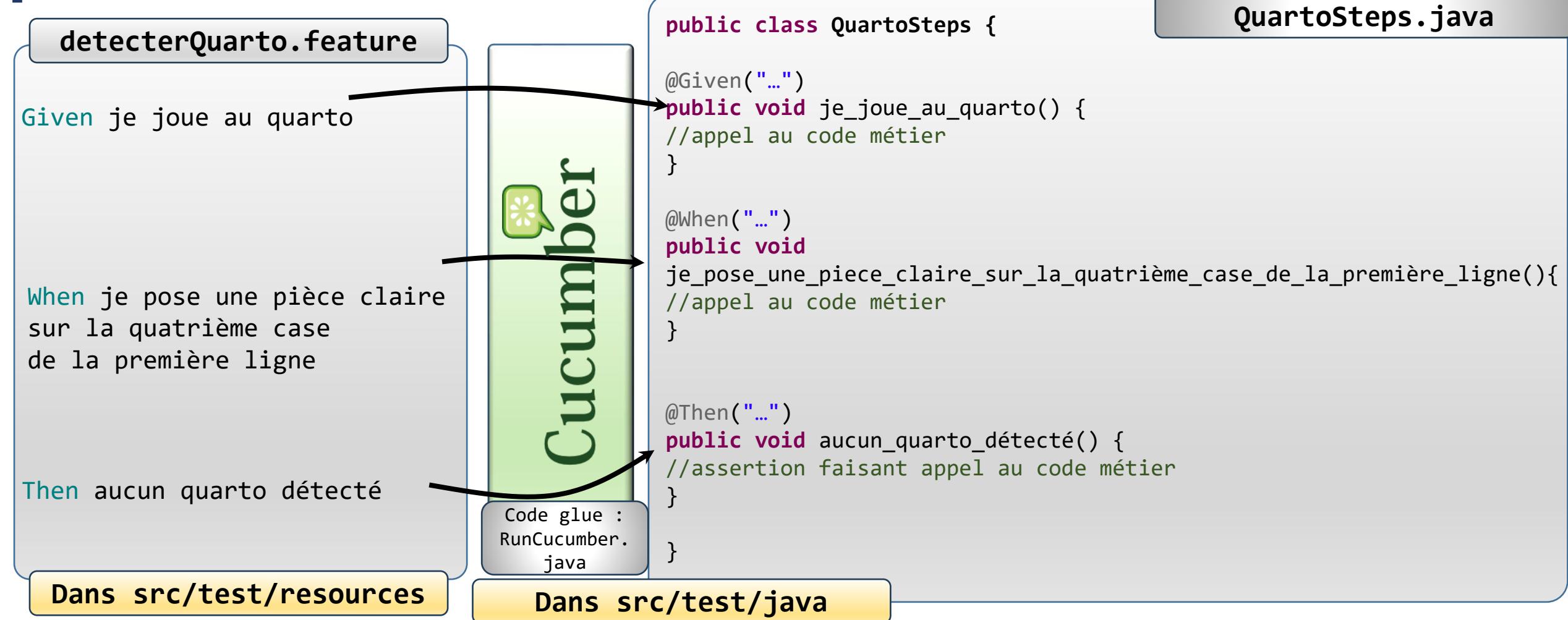
- Le scénario au cœur du langage Gherkin
- Ecrire un scénario paramétré, c'est possible ?
- Factoriser des étapes communes à différents scénarios, c'est possible ?
- Des scénarios, mais pourquoi ? ... pour décrire une fonctionnalité !
- Et où écrit-on le tout ? ... dans un document Gherkin !
- Des tags pour finir...
- Les yeux dans le code : de la conception à l'implémentation Java

# Quarto : Live Coding !

Vers un premier  
test d'acceptance  
en Double loop



# Principe de fonctionnement des frameworks BDD pour l'automatisation des tests d'acceptation



✓ Les outils de BDD permettent de traduire un scénario en langage naturel en appels de méthodes.

✓ La grammaire **Given/When/Then** (appelé langage **Gherkin**) permet de réaliser le mapping entre les « étapes » du scénario et les « steps » du code.



# Etape 1 : Ecrire le scénario en Gherkin

Dans src/test/resources/features

Feature: Détecer un quarto

DetectorQuarto.feature

En tant que joueur, je souhaite détecter un quarto afin de gagner la partie

Scenario: 4 pièces de même hauteur alignées sur une ligne

Given je joue au quarto

And il y a déjà de poser une pièce haute sur la première colonne de la deuxième ligne

And il y a déjà de poser une pièce haute sur la deuxième colonne de la deuxième ligne

And il y a déjà de poser une pièce haute sur la troisième colonne de la deuxième ligne

When je pose une pièce haute sur la quatrième colonne de la deuxième ligne

Then un quarto est détecté

<https://cucumber.io/docs/gherkin/>

# Cucumber

## Etape 2.1 : Code « glue » : RunCucumberTest pour la liaison step en gherkin - step en java

Dans src/test/java/features

```
package features;

import org.junit.platform.suite.api.ConfigurationParameter;
import org.junit.platform.suite.api.IncludeEngines;
import org.junit.platform.suite.api.SelectClasspathResource;
import org.junit.platform.suite.api.Suite;

import static io.cucumber.core.options.Constants.GLUE_PROPERTY_NAME;
import static io.cucumber.core.options.Constants.PLUGIN_PROPERTY_NAME;

@Suite
@IncludeEngines("cucumber")
@SelectClasspathResource("features")
@ConfigurationParameter(key = PLUGIN_PROPERTY_NAME, value = "pretty")
@ConfigurationParameter(key = GLUE_PROPERTY_NAME, value = "features")
public class RunCucumberTest {
```

RunCucumberTest.java

version JUnit 5

<https://cucumber.io/docs/cucumber/api/?lang=java#options>

# Etape 2.3 : Paramétrage `cucumber.properties` et `junit-platform.properties`

Affichage console à l'exécution de `RunCucumberTest.java`

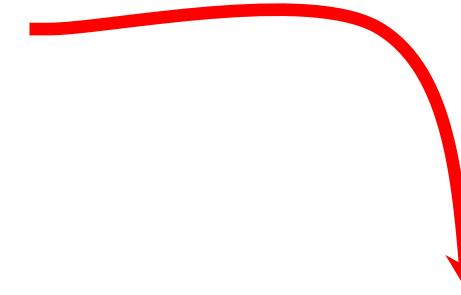
```
Share your Cucumber Report with your team at https://reports.cucumber.io
Activate publishing with one of the following:

src/test/resources/cucumber.properties:           cucumber.publish.enabled=true
src/test/resources/junit-platform.properties:       cucumber.publish.enabled=true
Environment variable:                            CUCUMBER_PUBLISH_ENABLED=true
JUnit:                                            @CucumberOptions(publish = true)

More information at https://cucumber.io/docs/cucumber/environment-variables/

Disable this message with one of the following:

src/test/resources/cucumber.properties:           cucumber.publish.quiet=true
src/test/resources/junit-platform.properties:       cucumber.publish.quiet=true
```



Dans `src/test/resources/features`

`cucumber.properties`

`cucumber.publish.quiet=true`  
`cucumber.publish.enabled=false`

`junit-platform.properties`

`cucumber.publish.quiet=true`  
`cucumber.publish.enabled=false`

# Etape 3.1 : Implémentation des steps gherkin en Java

Après l'exécution de **RunCucumberTest.java**, le code suivant est affiché dans la console,  
il n'y a plus qu'à la coller dans une classe java ...

Dans `src/test/java/features`

```
package features;
import io.cucumber.java.en.Given;
import io.cucumber.java.en.Then;
import io.cucumber.java.en.When;

public class QuartoSteps {

    @Given("je joue au quarto")
    public void je_joue_au_quarto() {
        // Write code here that turns the phrase above into concrete actions
        throw new io.cucumber.java.PendingException();
    }

    @Given("il y a déjà de poser une pièce haute sur la première colonne de la deuxième ligne")
    public void il_y_a_déjà_de_poser_une_pièce_haute_sur_la_première_colonne_de_la_deuxième_ligne() {
        // Write code here that turns the phrase above into concrete actions
        throw new io.cucumber.java.PendingException();
    }

    @Given("il y a déjà de poser une pièce haute sur la deuxième colonne de la deuxième ligne")
    public void il_y_a_déjà_de_poser_une_pièce_haute_sur_la_deuxième_colonne_de_la_deuxième_ligne() {
        // Write code here that turns the phrase above into concrete actions
        throw new io.cucumber.java.PendingException();
    }
}
```

QuartoSteps.java

# Etape 3.2 : Implémentation des steps gherkin en Java

Dans src/test/java/features

(suite)

QuartoSteps.java

```
@Given("il y a déjà de poser une pièce haute sur la troisième colonne de la deuxième ligne")
public void il_y_a_déjà_de_poser_une_pièce_haute_sur_la_troisième_colonne_de_la_deuxième_ligne() {
    // Write code here that turns the phrase above into concrete actions
    throw new io.cucumber.java.PendingException();
}

@When("je pose une pièce haute sur la quatrième colonne de la deuxième ligne")
public void je_pose_une_pièce_haute_sur_la_quatrième_colonne_de_la_deuxième_ligne() {
    // Write code here that turns the phrase above into concrete actions
    throw new io.cucumber.java.PendingException();
}

@Then("un quarto est détecté")
public void un_quarto_est_détecté() {
    // Write code here that turns the phrase above into concrete actions
    throw new io.cucumber.java.PendingException();
}
```

Write a  
failing  
acceptance  
test

# Etape 4 : Ajout d'un squelette de code métier pour faire échouer le test de la première boucle

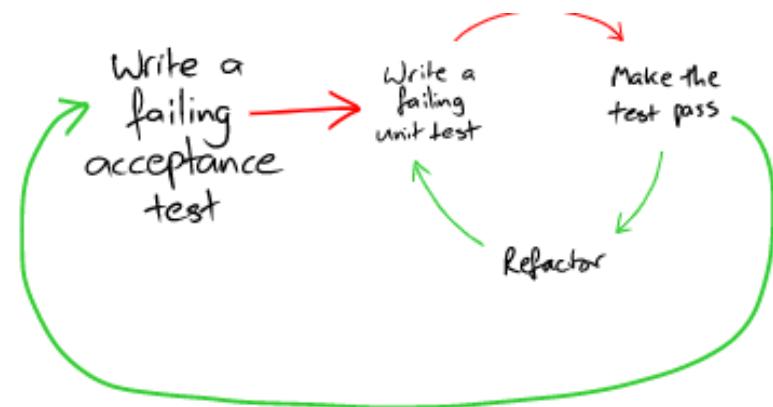
QuartoSteps.java

```
@Given("il y a déjà de poser une pièce haute sur la troisième colonne de la deuxième ligne")
public void il_y_a_déjà_de_poser_une_pièce_haute_sur_la_troisième_colonne_de_la_deuxième_ligne() {
    // Write code here that turns the phrase above into concrete actions
    throw new io.cucumber.java.PendingException();
}

@When("je pose une pièce haute sur la quatrième colonne de la deuxième ligne")
public void je_pose_une_pièce_haute_sur_la_quatrième_colonne_de_la_deuxième_ligne() {
    // Write code here that turns the phrase above into concrete actions
    throw new io.cucumber.java.PendingException();
}

@Then("un quarto est détecté")
public void un_quarto_est_détecté() {
    // Write code here that turns the phrase above into concrete actions
    throw new io.cucumber.java.PendingException();
}
```

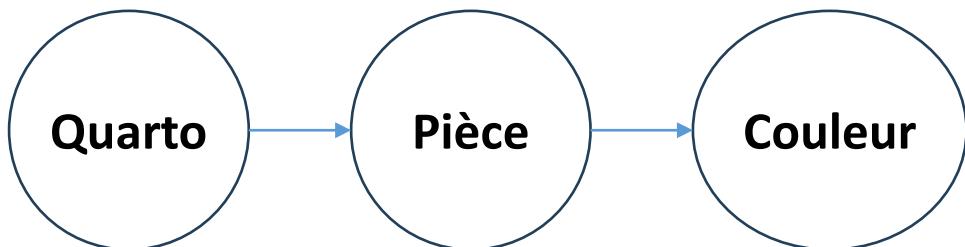
# Quarto : Double Loop En action !



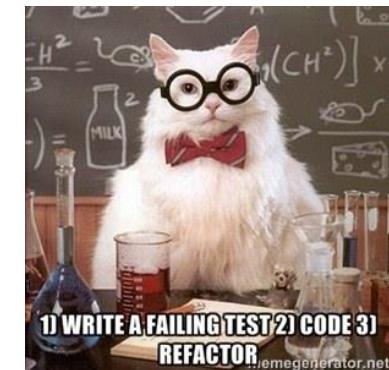
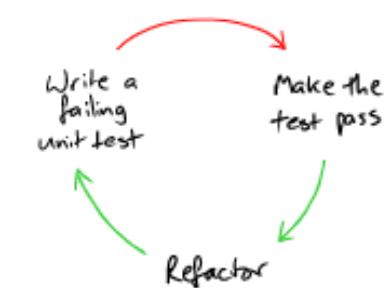
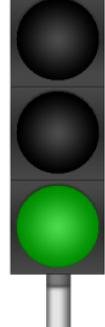
1. Faire compiler  
le test d'acceptance  
au rouge !



(avec un squelette de l'application)



2. Implémenter le métier  
en TDD pour faire passer  
le test d'acceptance !



# Un bootstrap d'un projet maven java pour Cucumber

This branch is 19 commits ahead, 40 commits behind swkBerlin:master.

- marc-bouvier ts-vitest: test defaults to watch, add other packagers
- .github/workflows Add Clojure CLI kata skeleton
- .scripts improve script
- clojure Add Clojure CLI kata skeleton
- cpp Extended documentation with docker
- csharp simplify working with dotnet6 starter
- d minimal -> a minimal
- elixir Delete config/config.exs for elixir skeleton
- fsharp fsharp
- golang ci for ginkgo
- haskell haskell stack test command
- java** java/kotlin : modify initial tests
- js explicit project names for deno
- kotlin java/kotlin : modify initial tests
- php make php works from docker
- python alias and rename to match default pattern
- rescript Bump ws from 7.3.1 to 7.5.7 in /rescript
- ruby/rspec gemfile lock
- rust these files should be committed
- scala env configuration
- swift Fix KataQuick resolved package version
- ts ts-vitest: test defaults to watch, add other packagers

<https://github.com/swcraftstras/kata-bootstraps/tree/master/java/cucumber>

github.com/swcraftstras/kata-bootstraps

Code Issues 1 Pull requests Actions Projects Security Insights

master kata-bootstraps / java / cucumber

marc-bouvier env configuration

This branch is 19 commits ahead of, 40 commits behind swkBerlin/kata-bootstraps:master .

Name	Last commit message
..	
docs	docu how to run features in IDEA
src/test	modern cucumber configuration
.gitignore	add support for cucumber jvm with junit 5
.sdkmanrc	env configuration
README.md	modern cucumber configuration
pom.xml	modern cucumber configuration

# A propos de la terminologie métier (ubiquitous language) du Quarto



Example Mapping  
favorise les  
discussions pour  
découvrir le métier



Etablir maintenant un glossaire sur  
la terminologie métier  
**(Ubiquitous language)**  
(ou vocable métier)

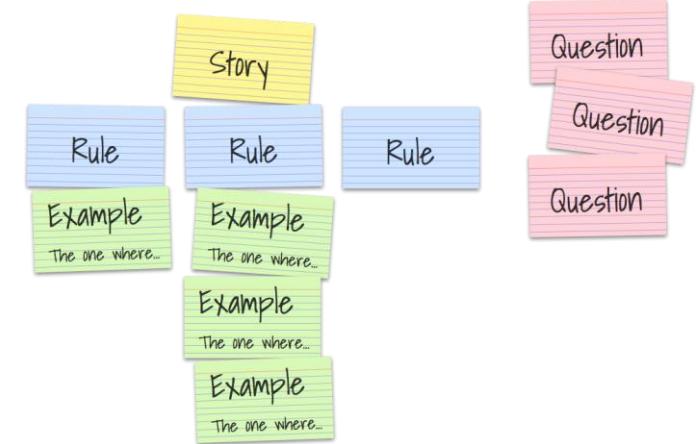
**terme** : définition (si temps)

..... ; ...

..... ; ...



Et si vous deviez modéliser l'UL du Quarto à partir de ce glossaire/termes métier, ça pourrait ressembler à quoi ?



# Example Mapping: Ressources !

## ***Article original***

<https://cucumber.io/blog/2015/12/08/example-mapping-introduction>

## ***Traduction en français***

<http://www.les-traducteurs-agiles.org/2017/03/21/presentation-cartographie-des-exemples.html>

<https://blog.ippon.fr/2020/07/15/example-mapping/>

## ***Des retours sur l'Example Mapping***

### **[Résumé Agile2015] Example Mapping :**

<https://medium.com/@fbourbonnais/r%C3%A9sum%C3%A9-agile2015-example-mapping-b1467e74189a#.y4l0up26y>

### **Example Mapping - Steering The Conversation:**

<http://blog.xebia.com/example-mapping-steering-the-conversation/>

### **Experiment with Example Mapping :**

<http://isacrispin.com/2016/06/02/experiment-example-mapping/>

# Annexe : A la découverte de l'Example Mapping



# Example Mapping : Mode d'emploi !

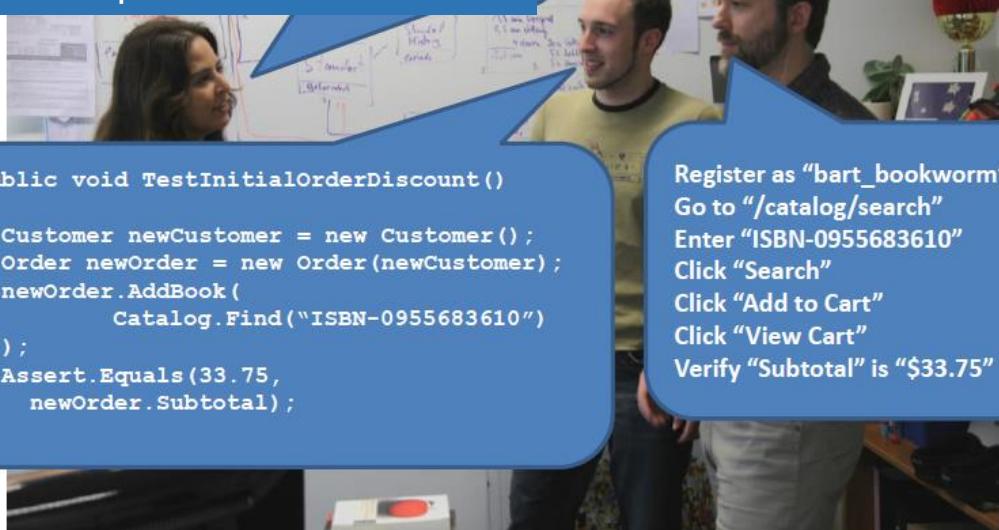


1. (D)écrire l'histoire à « raconter » sur un carte jaune et la placer au centre de la table ...
2. Noter chaque critère d'acceptance (ou règle) déjà connu(es) sur un carte bleue et les placer horizontalement en dessous de la story.
3. Pour illustrer chaque règle, proposer un ou plusieurs exemples sur un carte verte à déposer sous la règle.
4. La discussion autour des exemples peut soulever des questions pour lesquelles aucune des personnes présentes dans la pièce n'a la réponse. Noter les sur un carte rouge et poursuivre la discussion.

**Timebox:**  
**20-25 minutes max pour une story**  
(si plus semble nécessaire,  
vote pour savoir si story est prête  
ou nécessite encore discussion)

# Atelier des 3 amigos ...

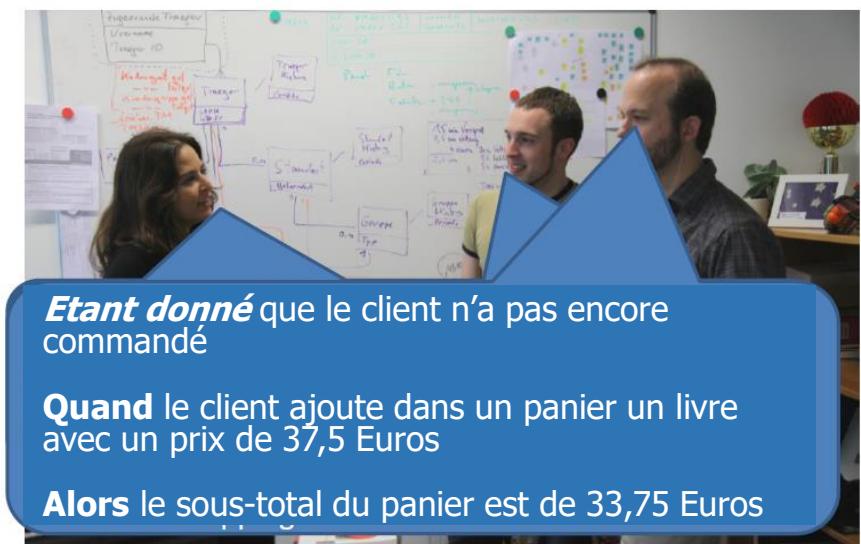
On désire encourager les nouveaux clients à réaliser un achat en leur offrant 10% sur leur premier achat



L'expression des besoins se fait en « **langage naturel** » (*Ubiquitous Language*)

La grammaire **Etant donné/Quand/Alors** (*Gherkin*) peut permettre de structurer le scénario

L'écriture des scénarios se fait en présentiel de **manière collaborative** (atelier des 3 Amigos)



# Et après ?

1. Les **Exemples** permettent d'**illustrer et bien comprendre les règles métiers** (exigences)

2. Les **Exemples** peuvent être transformés en **tests** si les 3 étapes *Arrange, Act et Assert* sont **jouées (déroulées)** de manière **manuelle ou automatique** afin d'obtenir **un verdict**.



3. Le verdict du **Test** permet de **vérifier & valider les exigences** du client (ses besoins)



# Tutoriel de prise en main de Cucumber



[https://github.com/iblasquez/tuto\\_bdd\\_cucumber](https://github.com/iblasquez/tuto_bdd_cucumber)

Attention ...

Tutoriel écrit en 2018 avec JUnit4



**Tutoriel de prise en main de Cucumber pour coder des tests Behavior Driven Development avec Java : recette aux concombres pour débutant**

**Table des matières**

- I. Installer le plug-in Cucumber-Eclipse
- II. Mettre en place votre premier projet Cucumber
  - II-A. Créer un projet Maven
  - II-B. Configurer le pom.xml pour Cucumber
  - II-C. Décrire le comportement en langage naturel (.feature dans src/test/resources)
  - II-D. Configurer le lanceur de test (dans src/test/java)
    - II-D-1. Crédit d'un fichier RunCucumberTest.java dans un package dojo de src/test/java
    - II-D-2. Exécution du lanceur de test (RunCucumberTest.java)
    - II-E. Implémenter le code de test des étapes (méthodes java) (dans src/test/java)
      - II-E-1. Le squelette des étapes issu du rapport de tests de la console
      - II-E-2. La configuration du lanceur de test pour une étape dite PENDING (option strict)
      - II-E-3. L'implémentation des étapes en Java pour produire le comportement attendu

Ce tutoriel s'inspire de l'exemple *Shakespeare du Kata Cucumber/Mockito* de Sébastien Mosser.

Cucumber est un framework de tests pour le « Behavior Driven Development », initialement développé en « Ruby », mais proposant aujourd'hui différentes implémentations pour de nombreux autres langages de programmation. Le site de référence est : [cucumber.io](http://cucumber.io).

Dans l'écosystème « Java », Cucumber est aujourd'hui l'un des frameworks BDD les plus utilisés.

Comme tout framework adapté au BDD, Cucumber permet de transformer les scénarios d'une « story » (écrits sous forme d'exemples en langage naturel au format Gherkin) en tests Java automatisés. Dans le principe, cette transformation est possible à l'aide du framework de tests JUnit. Chaque étape d'un scénario est implémentée comme une méthode Java, appelée step. Le lien entre la description textuelle de l'étape et le code Java de la step est réalisé via des annotations.

Dans ce tutoriel, nous verrons comment :

- I.Installer le plug-in Cucumber-Eclipse ;
- II.Mettre en place votre premier projet Cucumber :
  - o II-1. Créer un projet Maven,
  - o II-2. Configurer le pom.xml pour Cucumber,
  - o II-3. Décrire le comportement en langage naturel (.feature),
  - o II-4. Configurer le lanceur de tests,
  - o II-5. Implémenter le code de test des étapes (méthodes java),
  - o II-6. Implémenter le code métier de l'application.

Mais aussi, comment :