

# Spécification par l'exemple

## par l'exemple :

## le BDD démystifié

# Spécification par l'exemple par l'exemple : le BDD démystifié

- Introduction
- Constats
- Le BDD, c'est quoi ?
- Exemple complet sur une Bibliothèque
- Conclusion

# Introduction



But : donner envie d'essayer ou de réessayer le BDD

# Constats



Un PO lors de la démo du produit  
après une itération

# Constats



Les tests fonctionnels qui clignotent  
Les tests fonctionnels non exhaustifs

# Constats

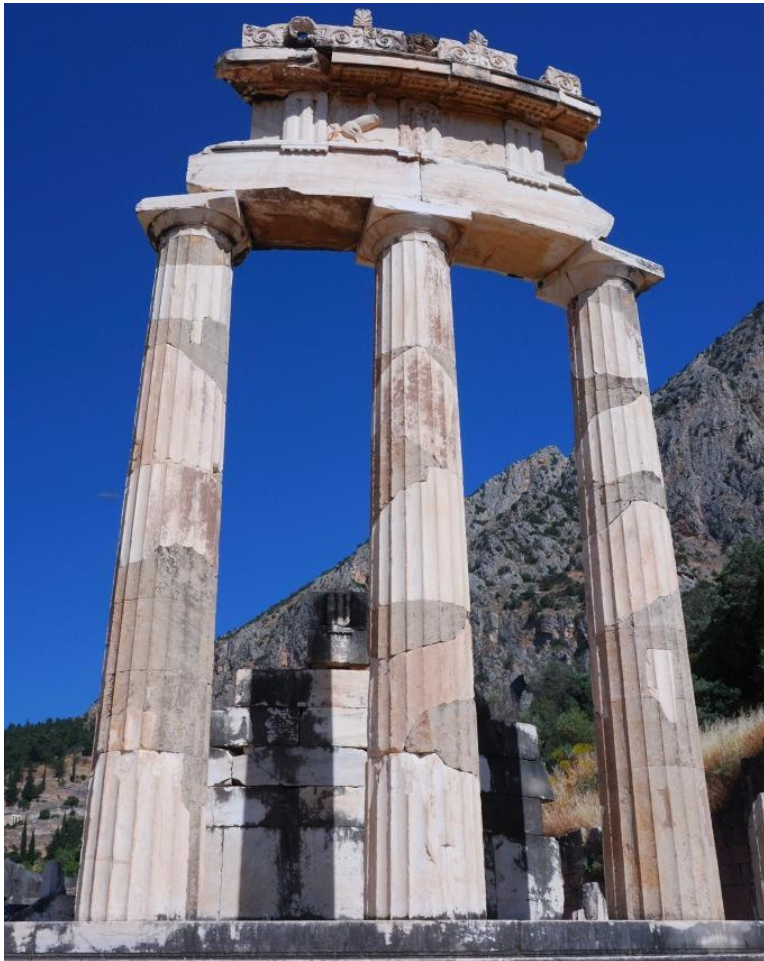


La documentation fonctionnelle  
après de nombreuses itérations  
et réajustement du besoin



# BDD, c'est quoi ?

## Behavior Driven Development



- Spécification par l'exemple
- Tests fonctionnels
- Documentation exécutable

# BDD, c'est quoi ?

## Spécification par l'exemple

**Scenario:** les suggestions proposées sont populaires, disponibles et adaptées à l'âge de l'utilisateur

**Etant donné** l'utilisateur "Tim"

**Et** il a "4" ans

**Et** les catégories populaires pour cet âge sont

categorieId	nom
cat1	Coloriage
cat2	Comptines

**Et** les livres disponibles pour les catégories "cat1,cat2" sont

livreId	titre	categorieId
lv11	Colorier les poules	cat1
lv21	Comptines de la ferme	cat2

**Quand** on demande "2" suggestions

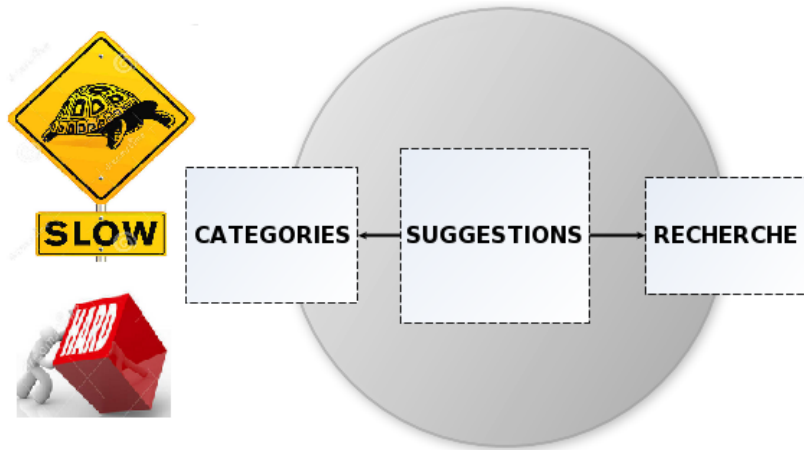
**Alors** les suggestions sont

livreId	titre	categorieId
lv11	Colorier les poules	cat1
lv21	Comptines de la ferme	cat2

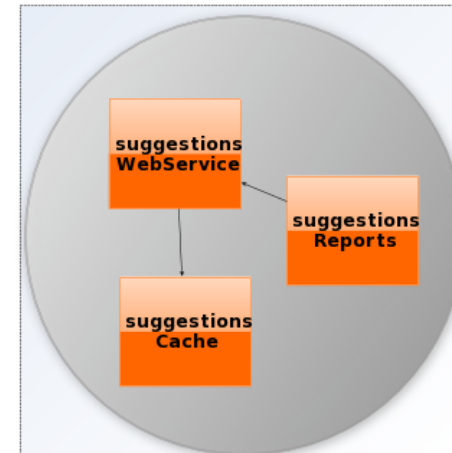


# BDD, c'est quoi ?

## Tests fonctionnels



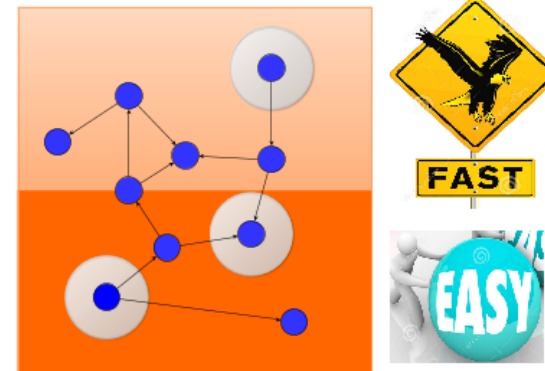
Entre systèmes



Sur un système  
Entre composants



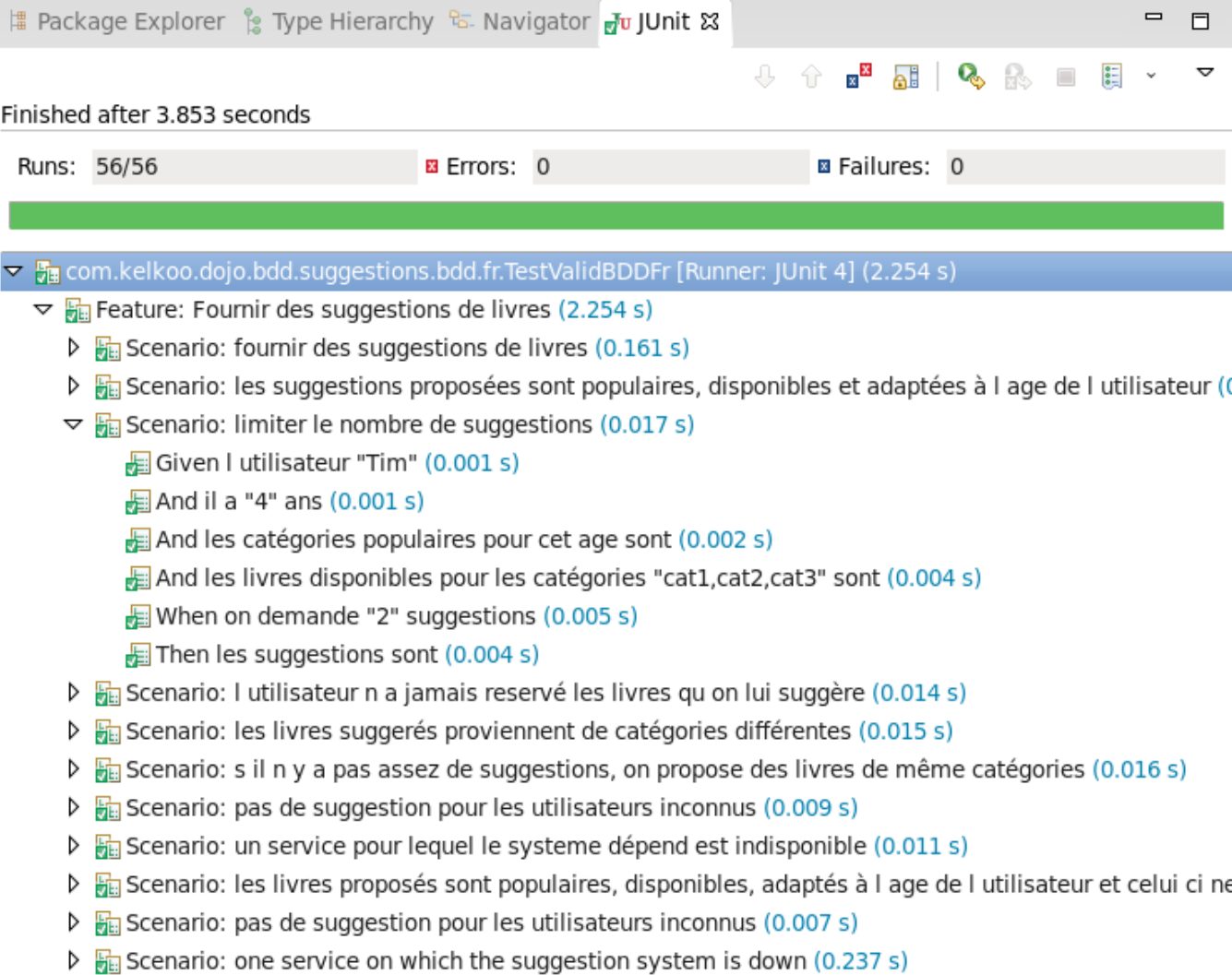
Sur un composant



Sur une classe

# BDD, c'est quoi ?

## Tests fonctionnels



The screenshot shows an IDE window with the JUnit tab selected. The test run is complete, having finished after 3.853 seconds. The summary bar indicates 56/56 runs, 0 errors, and 0 failures. The test suite is `com.kelkoo.dojo.bdd.suggestions.bdd.fr.TestValidBDDFr` [Runner: JUnit 4] (2.254 s). The test suite contains several scenarios, all of which passed. The first scenario is "Feature: Fournir des suggestions de livres (2.254 s)", which includes several sub-scenarios. The second scenario is "Scenario: limiter le nombre de suggestions (0.017 s)", which includes a "Given" block and a "When" block. The third scenario is "Scenario: l'utilisateur n'a jamais réservé les livres qu'on lui suggère (0.014 s)". The fourth scenario is "Scenario: les livres suggérés proviennent de catégories différentes (0.015 s)". The fifth scenario is "Scenario: s'il n'y a pas assez de suggestions, on propose des livres de même catégorie (0.016 s)". The sixth scenario is "Scenario: pas de suggestion pour les utilisateurs inconnus (0.009 s)". The seventh scenario is "Scenario: un service pour lequel le système dépend est indisponible (0.011 s)". The eighth scenario is "Scenario: les livres proposés sont populaires, disponibles, adaptés à l'âge de l'utilisateur et celui-ci ne (0.007 s)". The ninth scenario is "Scenario: pas de suggestion pour les utilisateurs inconnus (0.007 s)". The tenth scenario is "Scenario: one service on which the suggestion system is down (0.237 s)".

Package Explorer | Type Hierarchy | Navigator | JUnit

Finished after 3.853 seconds

Runs: 56/56 | Errors: 0 | Failures: 0

com.kelkoo.dojo.bdd.suggestions.bdd.fr.TestValidBDDFr [Runner: JUnit 4] (2.254 s)

- Feature: Fournir des suggestions de livres (2.254 s)
  - Scenario: fournir des suggestions de livres (0.161 s)
  - Scenario: les suggestions proposées sont populaires, disponibles et adaptées à l'âge de l'utilisateur (0.017 s)
  - Scenario: limiter le nombre de suggestions (0.017 s)
    - Given l'utilisateur "Tim" (0.001 s)
    - And il a "4" ans (0.001 s)
    - And les catégories populaires pour cet âge sont (0.002 s)
    - And les livres disponibles pour les catégories "cat1,cat2,cat3" sont (0.004 s)
    - When on demande "2" suggestions (0.005 s)
    - Then les suggestions sont (0.004 s)
  - Scenario: l'utilisateur n'a jamais réservé les livres qu'on lui suggère (0.014 s)
  - Scenario: les livres suggérés proviennent de catégories différentes (0.015 s)
  - Scenario: s'il n'y a pas assez de suggestions, on propose des livres de même catégorie (0.016 s)
  - Scenario: pas de suggestion pour les utilisateurs inconnus (0.009 s)
  - Scenario: un service pour lequel le système dépend est indisponible (0.011 s)
  - Scenario: les livres proposés sont populaires, disponibles, adaptés à l'âge de l'utilisateur et celui-ci ne (0.007 s)
  - Scenario: pas de suggestion pour les utilisateurs inconnus (0.007 s)
  - Scenario: one service on which the suggestion system is down (0.237 s)

# BDD, c'est quoi ?

## Documentation exécutable

**kelkoo** Spaces ▾ People Browse ▾ **Create**

Search

How-to articles

BDD as Behavior Driven Development

DOJO Library Suggestions - High level

**DOJO Library Suggestions - Specifications**

DOJO Library Suggestions - Technical Details

 Engineering / Platforms / Architecture / ... / BDD as Behavior Driven Development

**DOJO Library Suggestions - Specifications**

Created by Gerald Reinhart on Jul 21, 2016

Edit Watch

Documentation tested with success the last time: 27 Jul 2016, 14:34

Fournir des suggestions de livres

@level\_1\_specification @nominal\_case

Scenario: les suggestions proposées sont populaires, disponibles et adaptées à l'âge de l'utilisateur

Given l'utilisateur "Tim"

And il a "4" ans

And les catégories populaires pour cet âge sont

categoryId	categoryName
cat1	Coloriage
cat2	Comptines
cat3	Histoires pour le dodo

And les livres disponibles pour les catégories "cat1,cat2,cat3" sont

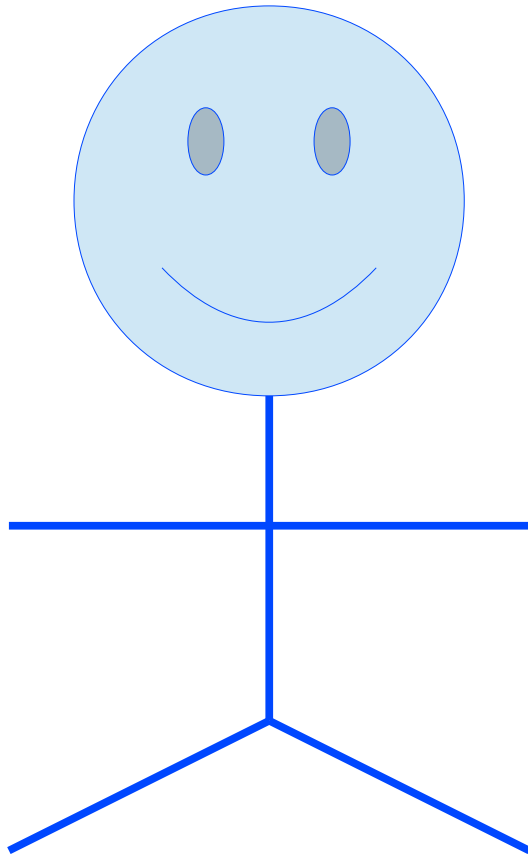
bookId	bookTitle	categoryId
b11	Colorier les poules	cat1
b21	Comptines de la ferme	cat2
b31	Histoires de la mer	cat3

When on demande "3" suggestions

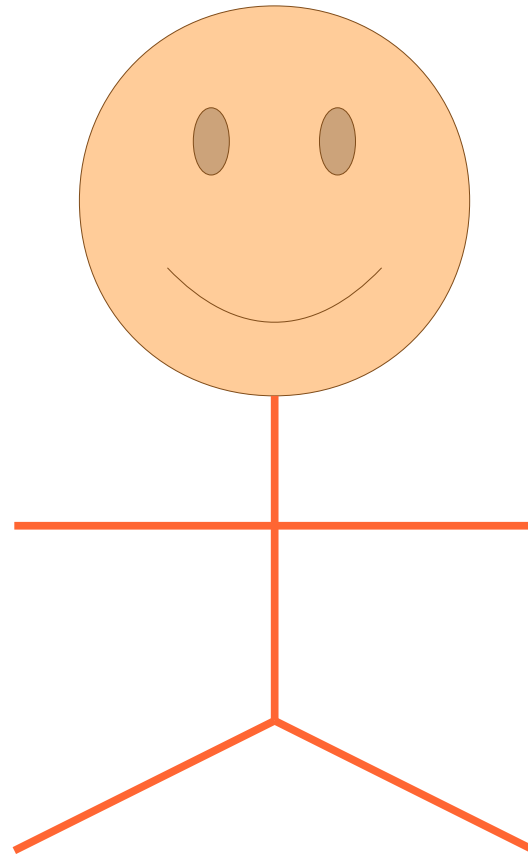
gerald.reinhart@kelkoo.com

# Bibliothèque

User Story à implémenter



PO



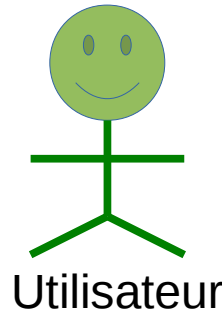
DEV

# Bibliothèque

User Story à implémenter

Catégorie

Les catégories des livres, catégories populaires par âge



Utilisateur

Utilisateur

Base utilisateurs, âge, livres lus...

Suggestion

Fournit des suggestions de livres

Recherche

Fournit des livres, recherche textuelle, recherche multi-critères (catégorie, popularité, disponibilité...)

Réservation

Réservation de livres, Livre disponible

# Bibliothèque

User Story à implémenter

**En tant qu'utilisateur de la bibliothèque,  
Je souhaite des suggestions de livres  
Afin de faire des découvertes**

## **Critères d'acceptance**

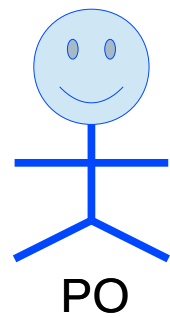
- Livre non lu par l'utilisateur
- Livre disponible



# Bibliothèque

User Story à implémenter

User Story



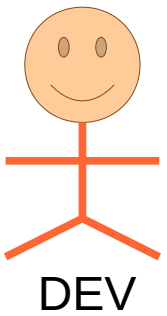
**En tant qu'**utilisateur de la bibliothèque, **Je souhaite** des suggestions de livres **Afin** de faire des découvertes

*Les suggestions doivent être adaptées à l'âge de l'utilisateur*

*Pour une meilleur découverte, les livres doivent venir de différentes catégories*

# Bibliothèque

User Story à implémenter



User Story

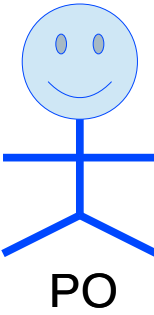
**En tant qu'**utilisateur de la bibliothèque, **Je souhaite** des suggestions de livres **Afin** de faire des découvertes

*Focalisé sur comment  
récupérer les livres, oublie  
que le livre doit être non lu  
par l'utilisateur*

*Manière la plus simple :  
faire une recherche sur  
la popularité des livres*

# Bibliothèque

## Écrire les scénarios



**Scenario:** fournir des suggestions de livres

**Etant donné** un utilisateur

**Quand** on demande suggestions

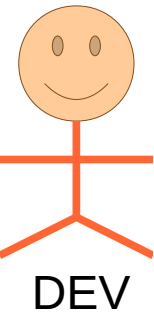
**Alors** les suggestions proposées sont populaires, disponibles et adaptées à l'âge de l'utilisateur

**Et** les suggestions proviennent de catégories différentes

*Manque : des exemples !*

# Bibliothèque

## Écrire les scénarios



**Scenario:** fournir des suggestions de livres

**Etant donné** l'utilisateur depuis le web service

<http://my.library.com/user/user1>

clé	valeur
userId	user1
age	4

*Un peu trop technique*

**Et** les livres depuis le web service

<http://my.library.com/search?popular=true&available=true>

livreId	titre	categorieId
lv11	livre11	cat1
lv21	livre21	cat2
lv31	livre31	cat3

*Exemple non parlant*

*Pas liés à l'utilisateur*

**Quand** on appelle

<http://localhost:9998/suggestions?userId=user1&maxResults=2>

**Alos** le code http retourné est "200"

**Et** les suggestions sont

livreId	titre	categorieId
lv11	livre11	cat1
lv21	livre21	cat2

*limite nombre suggestions*

# Bibliothèque

## Écrire les scénarios



**Scenario:** fournir des suggestions de livres

**Etant donné** l'utilisateur "Tim"

**Et** il a "4" ans

*Manque : limite nombre suggestions*

**Et** les catégories populaires pour cet age sont

categorieId	nom
cat1	Coloriage
cat2	Comptines
cat3	Histoires pour le dodo

*Manque : jamais lus*

**Et** les livres disponibles pour les catégories "cat1,cat2,cat3" sont

livreId	titre	categorieId
lv11	Colorier les poules	cat1
lv21	Comptines de la ferme	cat2
lv31	Histoires de la mer	cat3

**Quand** on demande "3" suggestions

**Alors** les suggestions sont

*Manque : catégories différentes*

livreId	titre	categorieId
lv11	Colorier les poules	cat1
lv21	Comptines de la ferme	cat2
lv31	Histoires de la mer	cat3

# Bibliothèque

## Écrire les scénarios



**Scenario:** fournir des suggestions de livres

**Etant donné** l'utilisateur "Tim"

**Et** il a "4" ans

**Et** les catégories populaires pour cet age sont

categorieId	nom
cat1	Coloriage
cat2	Comptines
cat3	Histoires pour le dodo

*Manque : jamais lus*

**Et** les livres disponibles pour les catégories "cat1,cat2,cat3" sont

livreId	titre	categorieId
lv11	Colorier les poules	cat1
lv21	Comptines de la ferme	cat2
lv31	Histoires de la mer	cat3

**Quand** on demande "2" suggestions

**Alors** les suggestions sont

livreId	titre	categorieId
lv11	Colorier les poules	cat1
lv21	Comptines de la ferme	cat2

*Manque : catégories différentes*

*limite nombre suggestions*



# Bibliothèque

## Écrire les scénarios



**Scenario:** fournir des suggestions de livres

**Etant donné** l'utilisateur "Tim"

*Manque : jamais lus*

**Et** il a "4" ans

**Et** les catégories populaires pour cet age sont

categorieId	nom
cat1	Coloriage
cat2	Comptines
cat3	Histoires pour le dodo

**Et** les livres disponibles pour les catégories "cat1,cat2,cat3" sont

livreId	titre	categorieId
lv11	Colorier les poules	cat1
lv12	Colorier les vaches	cat1
lv21	Comptines de la ferme	cat2
lv31	Histoires de la mer	cat3

*catégories différentes*

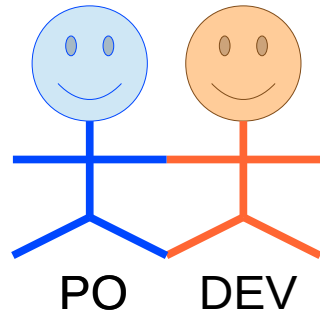
**Quand** on demande "2" suggestions

**Alors** les suggestions sont

livreId	titre	categorieId
lv11	Colorier les poules	cat1
lv21	Comptines de la ferme	cat2

# Bibliothèque

## Écrire les scénarios



**Scenario:** fournir des suggestions de livres

**Etant donné** l'utilisateur "Tim"

**Et** il a "4" ans

**Et** les catégories populaires po

catégorieId	nom
cat1	Coloriage
cat2	Comptines
cat3	Histoires pour le dodo

*On teste quoi au juste ?*

**Et** les livres disponibles pour les catégories "cat1,cat2,cat3" sont

livreId	titre	catégorieId
lv11	Colorier les poules	cat1
lv12	Colorier les vaches	cat1
lv13	Colorier les chevaux	cat1
lv21	Comptines de la ferme	cat2
lv31	Histoires de la mer	cat3

**Et** l'utilisateur a déjà réservé les livres suivants

livreId	titre	catégorieId
lv11	Colorier les poules	cat1

*jamais lus*

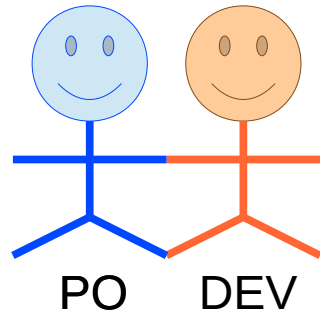
**When** on demande "2" suggestions

**Then** les suggestions sont

livreId	titre	catégorieId
lv12	Colorier les vaches	cat1
lv21	Comptines de la ferme	cat2

# Bibliothèque

## Écrire les scénarios



**Scenario:** les suggestions proposées sont populaires, disponibles et adaptées à l'âge de l'utilisateur

**Etant donné** l'utilisateur "Tim"

**Et** il a "4" ans

**Et** les catégories populaires pour cet âge sont

categorieId	nom
cat1	Coloriage
cat2	Comptines
cat3	Histoires pour le dodo

**Et** les livres disponibles pour les catégories "cat1,cat2,cat3" sont

livreId	titre	categorieId
lv11	Colorier les poules	cat1
lv21	Comptines de la ferme	cat2
lv31	Histoires de la mer	cat3

**Quand** on demande "3" suggestions

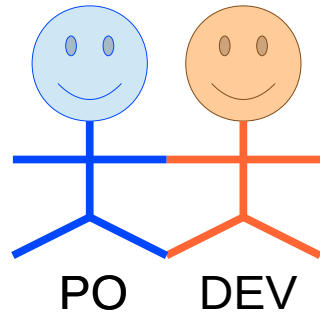
**Alors** les suggestions sont

livreId	titre	categorieId
lv11	Colorier les poules	cat1
lv21	Comptines de la ferme	cat2
lv31	Histoires de la mer	cat3

*Scenario 1 : cas nominal  
=> minimal*

# Bibliothèque

## Écrire les scénarios



**Scenario:** limiter le nombre de suggestions

Etant donné l'utilisateur "Tim"  
Et il a "4" ans  
Et les catégories populaires pour cet age sont

categorieId	nom
cat1	Coloriage
cat2	Comptines
cat3	Histoires pour le dodo

Et les livres disponibles pour les catégories "cat1,cat2,cat3" sont

livreId	titre	categorieId
lv11	Colorier les poules	cat1
lv21	Comptines de la ferme	cat2
lv31	Histoires de la mer	cat3

When on demande "2" suggestions

Then les suggestions sont

livreId	titre	categorieId
lv11	Colorier les poules	cat1
lv21	Comptines de la ferme	cat2

*Scenario 2 : cas nominal*

*Simplifions encore le scenario*

# Bibliothèque

## Écrire les scénarios



**Scenario:** limiter le nombre de suggestions

**Etant donné** un utilisateur

**Et** "3" livres sont disponibles pour  
les catégories populaires pour cet age

**When** on demande "2" suggestions

**Then** "2" suggestions sont proposées  
parmi les livres précédents

*Scenario 2 : cas nominal*

# Bibliothèque

## Écrire les scénarios



**Scenario:** l'utilisateur n'a jamais réservé les livres qu'on lui suggère

### Scenario 3 : cas nominal

Etant donné l'utilisateur "Tim"  
Et il a "4" ans  
Et les catégories populaires pour cet age sont

catégorieId	nom
cat1	Coloriage
cat2	Comptines
cat3	Histoires pour le dodo

Et les livres disponibles pour les catégories "cat1,cat2" sont

livreId	titre	catégorieId
lv11	Colorier les poules	cat1
lv21	Comptines de la ferme	cat2

Et l'utilisateur a déjà réservé les livres suivants

livreId	titre	catégorieId
lv11	Colorier les poules	cat1

Quand on demande "1" suggestions

Alors les suggestions sont

livreId	titre	catégorieId
lv21	Comptines de la ferme	cat2

Déroulons l'algo



# Bibliothèque

## Écrire les scénarios



**Scenario:** les livres suggérés proviennent de catégories différentes

```
Etant donné l'utilisateur "Tim"
Et il a "4" ans
Et les catégories populaires pour cet age sont
  | categorieId | nom                |
  | cat1       | Coloriage          |
  | cat2       | Comptines          |
  | cat3       | Histoires pour le dodo |
```

*Scenario 4 : cas nominal*

```
Et les livres disponibles pour les catégories "cat1,cat2" sont
  | livreId | titre                | categorieId |
  | lv11    | Colorier les poules  | cat1       |
  | lv12    | Colorier les vaches  | cat1       |
  | lv21    | Comptines de la ferme | cat2       |
```

Quand on demande "2" suggestions

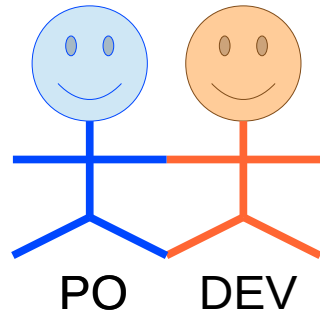
Alors les suggestions sont

```
  | livreId | titre                | categorieId |
  | lv11    | Colorier les poules  | cat1       |
  | lv21    | Comptines de la ferme | cat2       |
```

*Déroulons l'algo*

# Bibliothèque

## Écrire les scénarios



**Scenario:** s'il n y a pas assez de suggestions,  
on propose des livres de même catégories

### Scenario 5 : cas limite

**Etant donné** l'utilisateur "Tim"

**Et** il a "4" ans

**Et** les catégories populaires pour cet age sont

categorieId	nom
cat1	Coloriage
cat2	Comptines

**Et** les livres disponibles pour les catégories "cat1,cat2" sont

livreId	titre	categoryId
lv11	Colorier les poules	cat1
lv12	Colorier les vaches	cat1
lv21	Comptines de la ferme	cat2

**Quand** on demande "3" suggestions

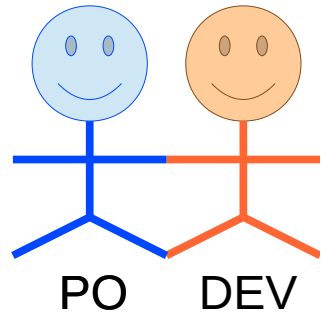
**Alors** les suggestions sont

livreId	titre	categoryId
lv11	Colorier les poules	cat1
lv21	Comptines de la ferme	cat2
lv12	Colorier les vaches	cat1

Déroulons l'algo

# Bibliothèque

## Écrire les scénarios



**Scenario:** pas de suggestion pour  
les utilisateurs inconnus

**Etant donné** l'utilisateur "Lise"

**Et** l'utilisateur est inconnu

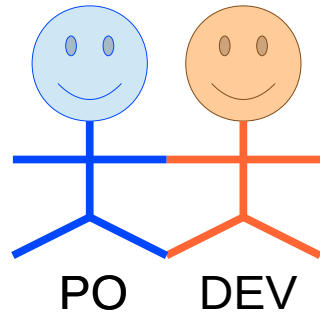
**Quand** on demande "3" suggestions

**Alors** aucune suggestion est proposée

*Scenario 6 : cas limite*

# Bibliothèque

## Écrire les scénarios



**Scenario:** un service pour lequel le système dépend est indisponible

**Etant donné** l'utilisateur "Tim"

**Et** impossible de récupérer les informations de l'utilisateur

**Quand** on demande "3" suggestions

**Alors** le système est temporairement indisponible

*Scenario 7 : cas d'erreur*

# Bibliothèque

## Écrire les scénarios



**Scenario:** les suggestions proposées sont populaires, disponibles et adaptées à l'âge de l'utilisateur

**Etant donné** l'utilisateur depuis le web service <http://my.library.com/user/Tim>

clé	valeur
userId	Tim
age	4

### Scenario 1 version technique

**Et** les catégories depuis le web service

<http://my.library.com/category?popular=true&age=4>

categorieId	nom
cat1	Coloriage
cat2	Comptines
cat3	Histoires pour le dodo

**Et** les livres depuis le web service

<http://my.library.com/search?categories=cat1,cat2,cat3&available=true>

livreId	titre	categorieId
lv11	Colorier les poules	cat1
lv21	Comptines de la ferme	cat2
lv31	Histoires de la mer	cat3

**Et** les livres depuis le web service <http://my.library.com/user/Tim/books>

livreId	titre	categorieId
lv11	Colorier les poules	cat1

**Quand** on appelle <http://localhost:9998/suggestions?userId=Tim&maxResults=3>

**Alors** le code http retourné est "200"

**Et** les suggestions sont

livreId	titre	categorieId
lv11	Colorier les poules	cat1
lv21	Comptines de la ferme	cat2
lv31	Histoires de la mer	cat3

# Bibliothèque

Organiser les scénarios



**En tant qu'utilisateur de la bibliothèque,  
Je souhaite** des suggestions de livres  
**Afin** de faire des découvertes

*Scenario 1*

*Scenario 2*

*Scenario 7*

*Scenario 0*

*Scenario 5*

*Scenario 3*

*Scenario 4*

*Scenario 6*

*Scenario 7*

*Scenario 1*

*Scenario 6*



# Bibliothèque

Organiser les scénarios



**En tant qu'utilisateur de la bibliothèque,**  
**Je souhaite** des suggestions de livres  
**Afin** de faire des découvertes

@nominal\_case

@limit\_case

@error\_case

@level\_0\_  
high\_level

*Scenario 0*

@level\_1\_  
specification

*Scenario 1*

*Scenario 2*

*Scenario 3*

*Scenario 4*

*Scenario 5*

*Scenario 6*

*Scenario 7*

@level\_2\_  
technical

*Scenario 1*

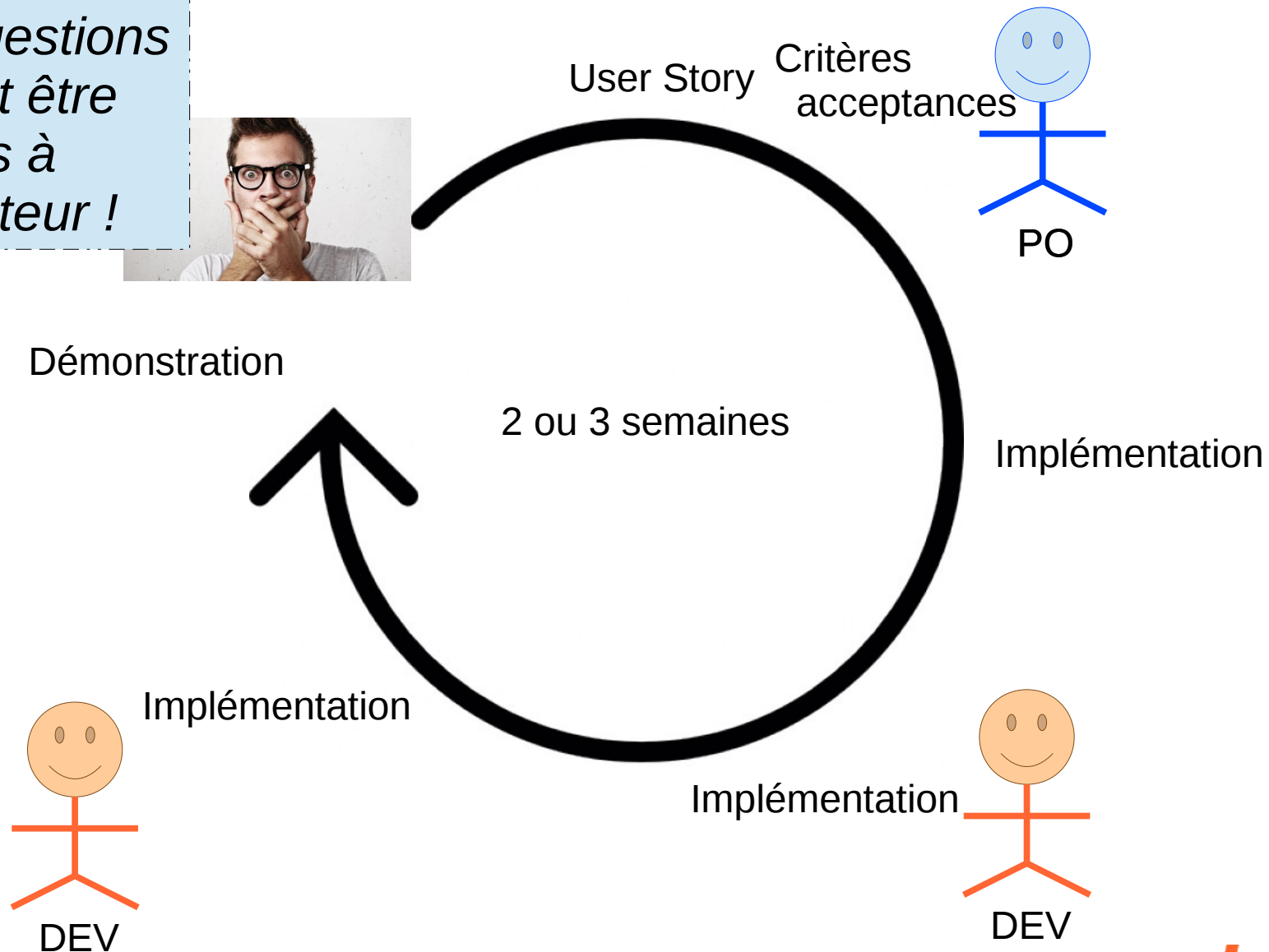
*Scenario 6*

*Scenario 7*

# Bibliothèque

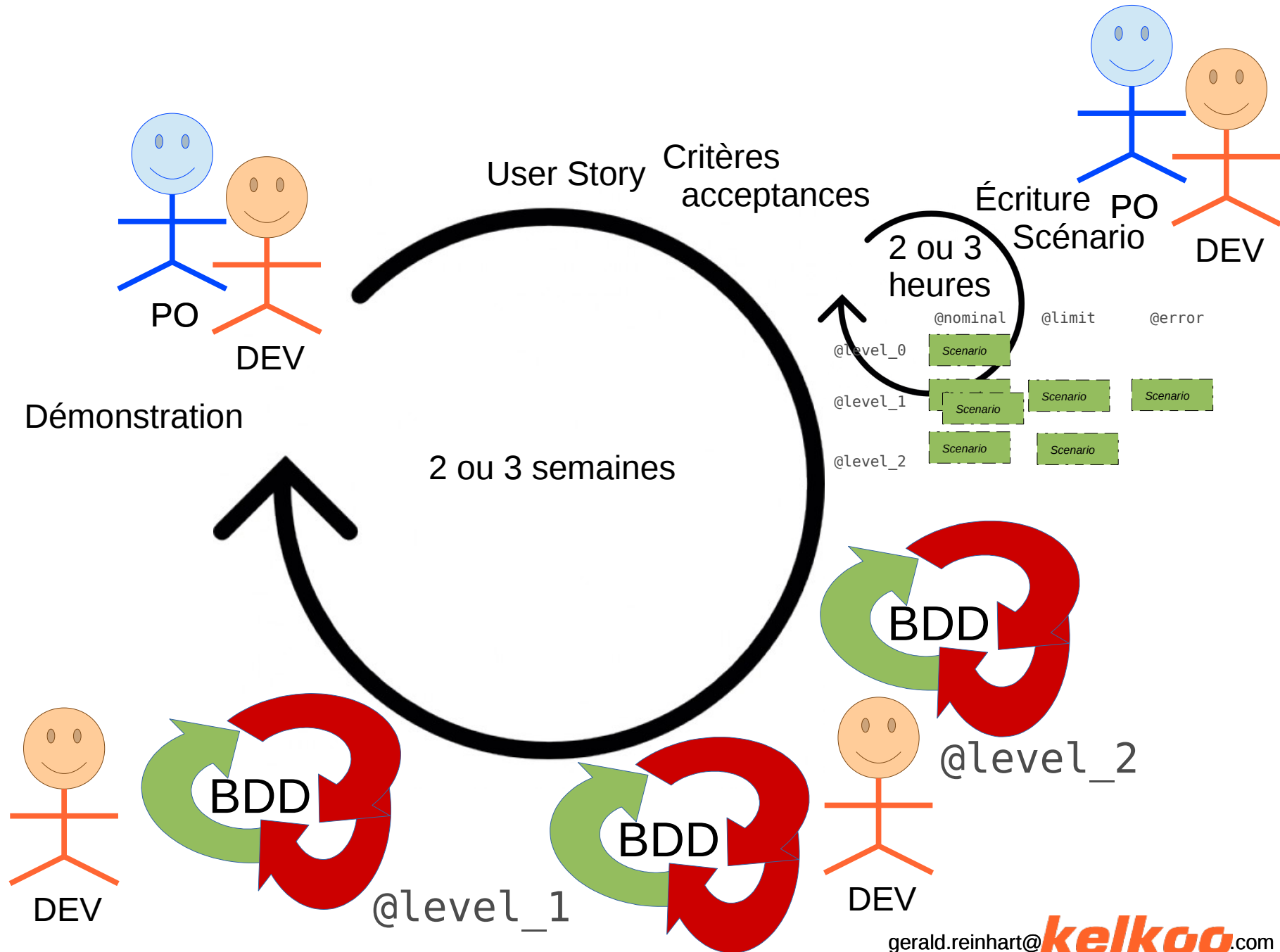
Sans spécification par l'exemple

*Les suggestions  
doivent être  
liées à  
l'utilisateur !*



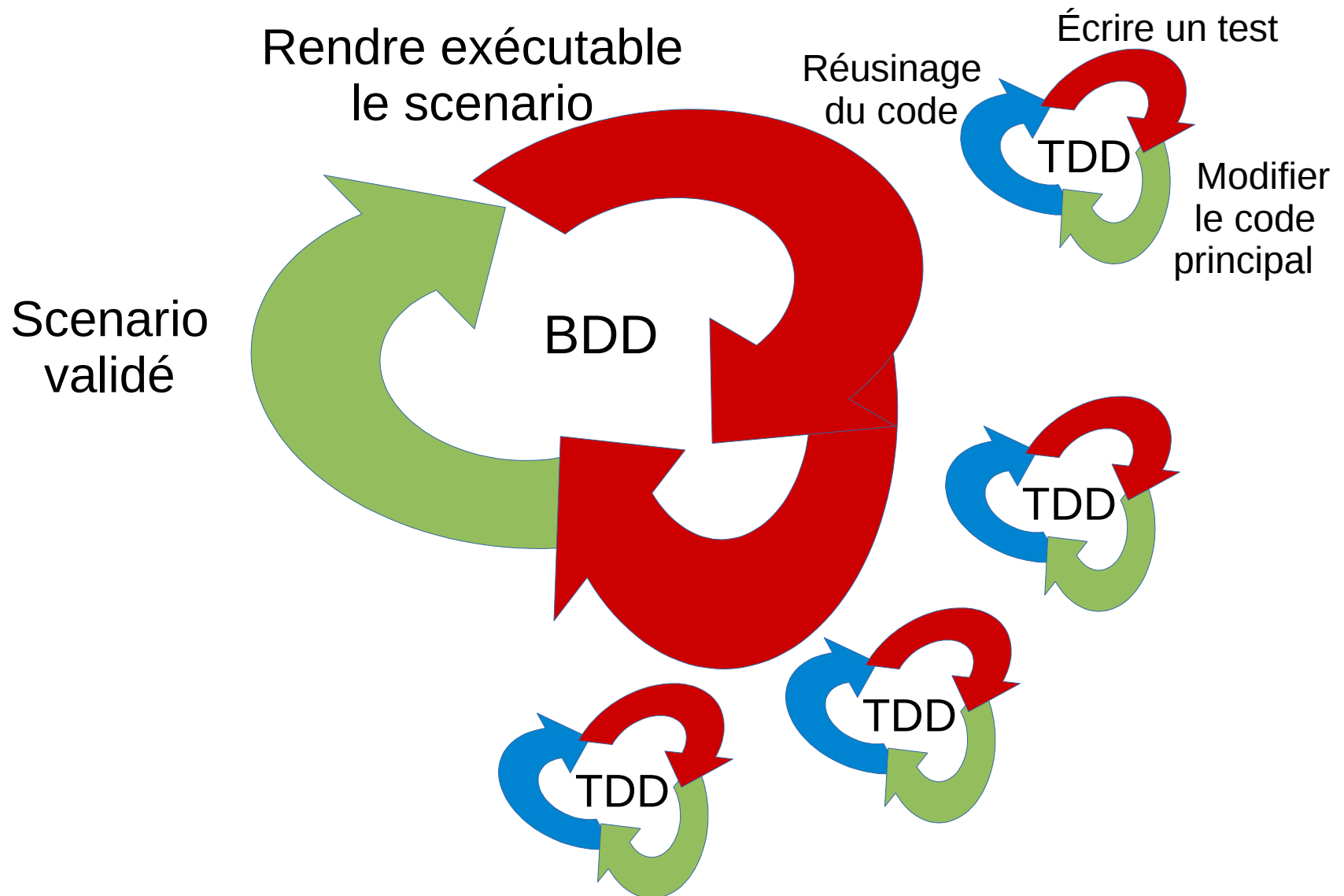
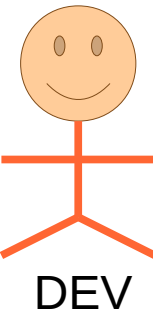
# Bibliothèque

## Feuille de route



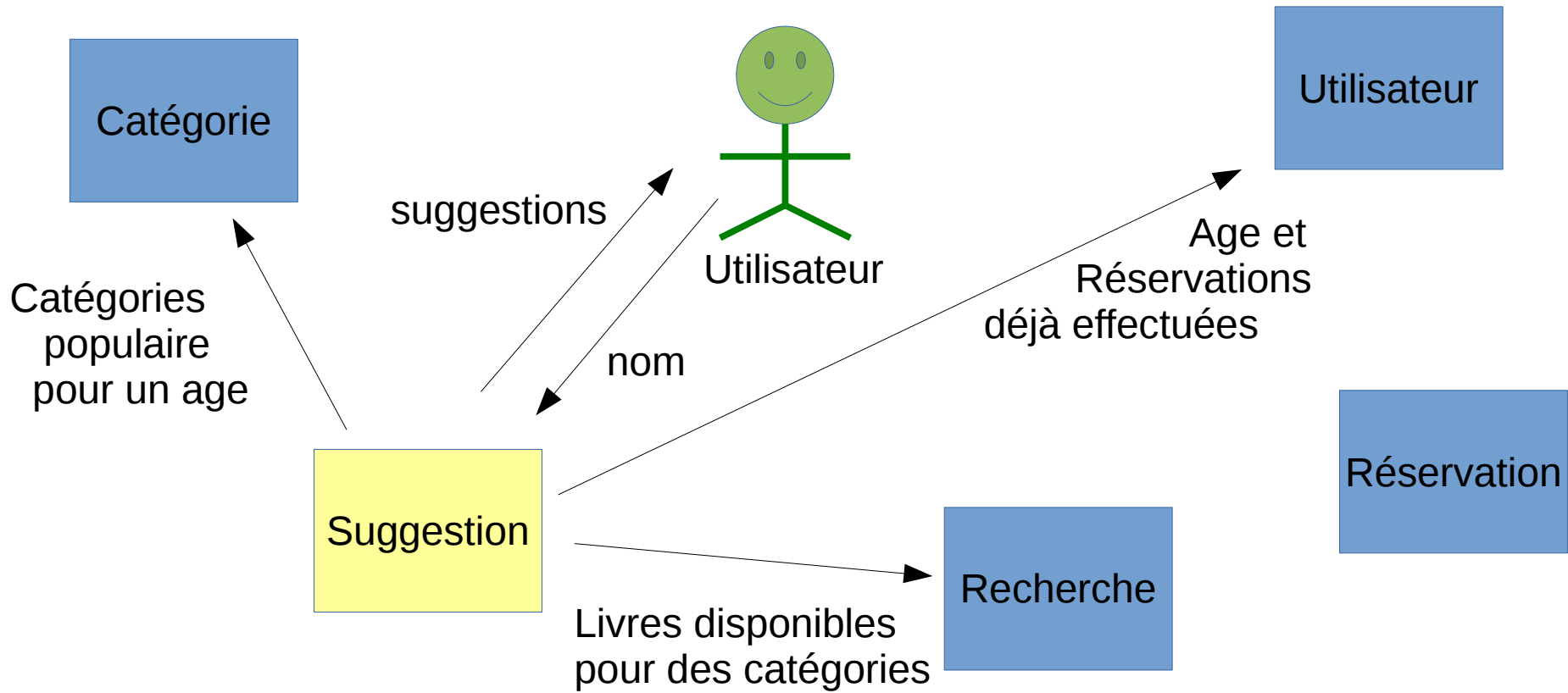
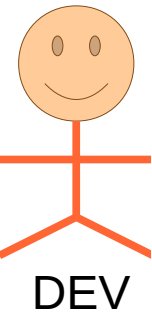
# Bibliothèque

Rendre exécutable les scénarios



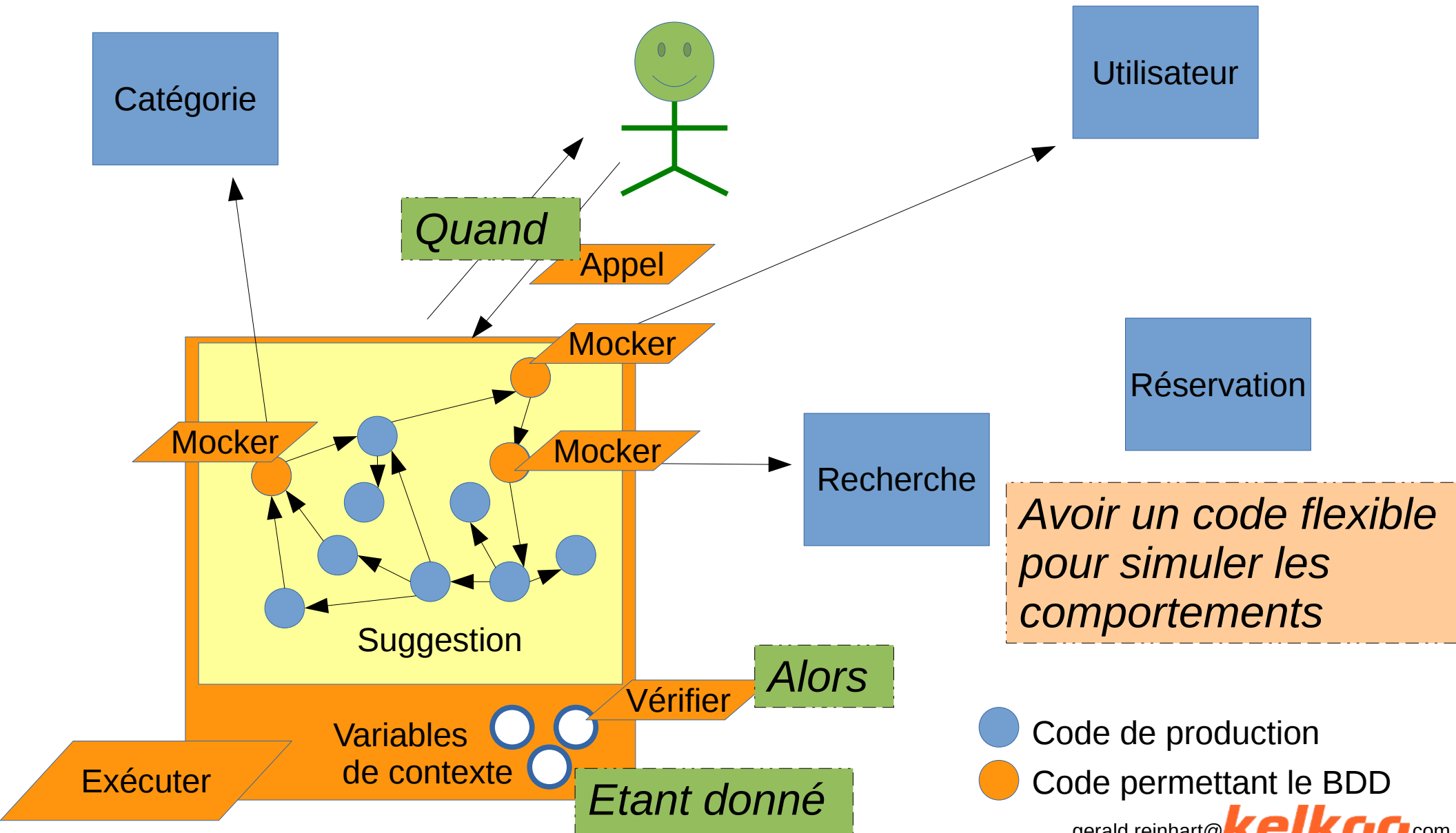
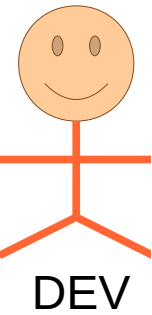
# Bibliothèque

Rendre exécutable les scénarios



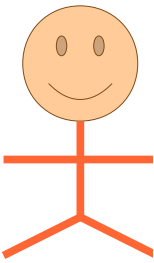
# Bibliothèque

Rendre exécutable les scénarios



# Bibliothèque

Rendre exécutable les scénarios



```
@level_2_technical_details @nominal_case @ongoing
```

**Scenario:** les livres proposés sont populaires, disponibles, adaptés à l'âge de l'utilisateur

**Given** l'utilisateur depuis le web service `http://my.library.com/user/Tim`

field	value
userId	Tim
age	4

**And** les catégories depuis le web service

categoryId	categoryName
cat1	Coloriage
cat2	Comptines

Runs: 0/7

Errors: 14

Failures: 0

com.kelkoo.dojo.bdd.suggestions.bdd.fr.OnGoingBDDFrTest [Runner: JUnit 4]

Feature: Fournir des suggestions de livres

Scenario: les livres proposés sont populaires, disponibles, adaptés à l'âge de l'utilisateur

Given l'utilisateur depuis le web service `http://my.library.com/user/Tim`

And les catégories depuis le web service `http://my.library.com/category?popular=true`

And les livres depuis le web service `http://my.library.com/search?categories=cat1,cat2`

And les livres depuis le web service `http://my.library.com/user/Tim/books`

When on appelle `http://localhost:9998/suggestions?userId=Tim&maxResults=3`

Then le code http retourné est "200"

Then les suggestions sont

Problems @ Javadoc Search Outline Task List Progress Console

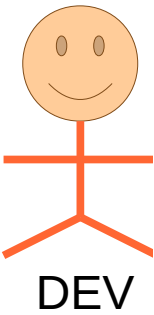
*Le développeur est complètement guidé*

You can implement missing steps with the snippets below:

```
@Given("^l utilisateur depuis le web service http://my.library.com/user/Tim$")
public void l_utilisateur_depuis_le_web_service_http_my_library_com_user_Tim(DataTable arg1) throws Throwable {
    // Express the Regexp above with the code you wish you had
    // For automatic conversion, change DataTable to List<YourType>
    throw new PendingException();
}
```

# Bibliothèque

Rendre exécutable les scénarios



*Pont entre l'étape du scénario et le code*

```
@Given("^l utilisateur depuis le web service http://my.library.com/user/([^\""]*)$")
public void given_the_user_from_user_ws(String userId, List<FieldValue> values) throws Throwable {
    FieldValues fieldsValues = new FieldValues(values);
    user.setUserId(userId);
    user.setAge(fieldsValues.getAsInteger("age"));
    when(usersWSClientMock.retrieveUser(user.getUserId())).thenReturn(user);
}
```

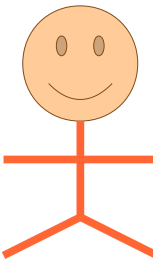
*Variables de contexte*

(x)= Variables ⌘ Breakpoints ⌘ Expressions	
Name	Value
▶ user	User (id=91)
User [userId=Tim, age=4, alreadyBookedBooks=[]]	



# Bibliothèque

Rendre exécutable les scénarios



```
com.kelkoo.dojo.bdd.suggestions.bdd.fr.Or
Feature: Fournir des suggestions de livre
Scenario: les livres proposés sont pop
  Given l utilisateur depuis le web se
  And les catégories depuis le web s
  And les livres depuis le web servic
  And les livres depuis le web servic
  When on appelle http://localhost:9
  Then le code http retourné est "20
  Then les suggestions sont
```

```
@level_2_technical_details @nominal_case @ongoing
```

```
Scenario: les livres proposés sont populaires, disponibles, adaptés à l age de l utilisateur
```

```
Given l utilisateur depuis le web service http://my.library.com/user/Tim
```

field	value
userId	Tim
age	4

```
And les catégories depuis le web service http://my.library.com/category?popular=true&age=4
```

categoryId	categoryName
cat1	Coloriage
cat2	Comptines
cat3	Histoires pour le dodo

```
And les livres depuis le web service http://my.library.com/search?categories=cat1,cat2,cat3
```

bookId	bookTitle	categoryId
b11	Colorier les boules	cat1

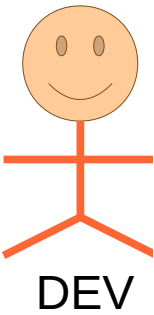
## Définition du comportement des mocks

```
@Given("^les catégories depuis le web service http://my.library.com/category\\?popular=([^\"]*)&age=(\\d+)$")
public void given_the_categories_from_categories_ws(Boolean popular, Integer age, List<Category> popularCategoriesGivenAgeUser) throws Exception {
    when(categoriesWSClientMock.retrieveCategories(popular, user.getAge())).thenReturn(popularCategoriesGivenAgeUser);
}
```

```
com.kelkoo.dojo.bdd.suggestions.bdd.fr.OnGoingBDDFrTest [Runner: JUnit 4] (0.010 s)
Feature: Fournir des suggestions de livres (0.010 s)
Scenario: les livres proposés sont populaires, disponibles, adaptés à l age de l utilisateur
  Given l utilisateur depuis le web service http://my.library.com/user/Tim (0.004 s)
  And les catégories depuis le web service http://my.library.com/category?popular=true
  And les livres depuis le web service http://my.library.com/search?categories=cat1,cat2,cat3
  And les livres depuis le web service http://my.library.com/user/Tim/books
```

# Bibliothèque

## Rendre exécutable les scénarios



- ▼ com.kelkoo.dojo.bdd.suggestions.bdd.fr.OnGoingBDDFrTest [Runner: JUnit 4] (0.019 s)
  - ▼ Feature: Fournir des suggestions de livres (0.019 s)
    - ▼ Scenario: les livres proposés sont populaires, disponibles, adaptés à l'âge de l'utilisateur
      - Given l'utilisateur depuis le web service http://my.library.com/user/Tim (0.004 s)
      - And les catégories depuis le web service http://my.library.com/category?popular=true
      - And les livres depuis le web service http://my.library.com/search?categories=cat1,cat2
      - And les livres depuis le web service http://my.library.com/user/Tim/books (0.003 s)
      - When on appelle http://localhost:9998/suggestions?userId=Tim&maxResults=3
      - Then le code http retourné est "200"

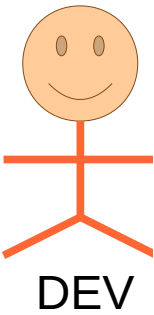
```
@When("^on appelle ([^\\"]*)$")
public void when_we_call_suggestions_ws(String suggestionsUrl) throws Throwable {
    wsSuggestionsResponse = client.resource(suggestionsUrl).accept("application/xml").get(ClientResponse.class);
}
```

*Activation du code réel*

- ▼ com.kelkoo.dojo.bdd.suggestions.bdd.fr.OnGoingBDDFrTest [Runner: JUnit 4] (0.101 s)
  - ▼ Feature: Fournir des suggestions de livres (0.101 s)
    - ▼ Scenario: les livres proposés sont populaires, disponibles, adaptés à l'âge de l'utilisateur
      - Given l'utilisateur depuis le web service http://my.library.com/user/Tim (0.005 s)
      - And les catégories depuis le web service http://my.library.com/category?popular=true
      - And les livres depuis le web service http://my.library.com/search?categories=cat1,cat2
      - And les livres depuis le web service http://my.library.com/user/Tim/books (0.080 s)
      - When on appelle http://localhost:9998/suggestions?userId=Tim&maxResults=3 (0.001 s)
      - Then le code http retourné est "200"
      - Then les suggestions sont

# Bibliothèque

Rendre exécutable les scénarios



```
@Then("^le code http retourné est \"([^\"]*)\"$")
public void the_http_code_is(Integer httpCode) throws Throwable {
    assertThat(wsSuggestionsResponse.getStatus(), is(httpCode));
}
```

When on appelle http://localhost:9998/suggestions?userId=Tim&maxResults=3 (0.002 s)

Then le code http retourné est "200" (0.002 s)

Then les suggestions sont

Failure Trace

```
java.lang.AssertionError:
    Expected: is <200>
    but: was <405>
    at org.hamcrest.MatcherAssert.assertThat(MatcherAssert.java:20)
```

*Le code principal n'existe pas....*

```
@GET
@Produces("application/xml")
public Suggestions getSuggestions(@QueryParam("userId") String userId, @QueryParam("maxResults") Integer maxResults) {
    return new Suggestions();
}
```

When on appelle http://localhost:9998/suggestions?userId=Tim&maxResults=3 (0.003 s)

Then le code http retourné est "200" (0.003 s)

Then les suggestions sont

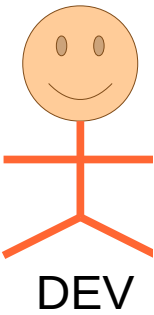
Failure Trace

```
cucumber.api.PendingException: TODO: implement
    at cucumber.runtime.junit4.JUnit4Reporter.addFailureReport(JUnit4Reporter.java:71)
```

*Vérification du résultat*

# Bibliothèque

## Implémenter les scénarios



```
@Then("^les suggestions sont$")
public void then_the_suggestions_are(List<Suggestion> expectedSuggestions) throws Throwable {
    SuggestionsMarshaller suggestionsMarshaller = new SuggestionsMarshaller();
    Suggestions actualSuggestions = suggestionsMarshaller.deserialize(wsSuggestionsResponse.getEntity(String.class));
    checkSameSuggestions(actualSuggestions, expectedSuggestions);
}
```

Then le code http retourné est "200" (0.012 s)

Then les suggestions sont (0.002 s)

Failure Trace

java.lang.AssertionError:

Expected: <2>

but: was <0>

at org.hamcrest.MatcherAs

*Écrivons réellement le code*

@GET

@Produces("application/xml")

```
public Suggestions getSuggestions(@QueryParam("userId") String userId, @QueryParam("maxResults") Integer maxResults) {
```

```
    Suggestions suggestions = new Suggestions();
```

```
    maxResults = maxResults == null ? DEFAULT_MAX_RESULT : maxResults;
```

```
    User user = userWSClient.retrieveUser(userId);
```

```
    Boolean isPopular = true;
```

```
    List<Category> popularCategories = categoriesWSClient.retrieveCategories(isPopular, user.getAge());
```

```
    Boolean bookAvailable = true;
```

```
    List<Book> books = searchWSClient.searchBooks(bookAvailable, extractCategoryIds(popularCategories));
```

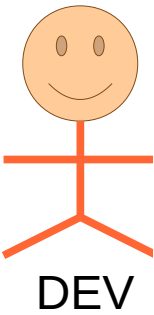
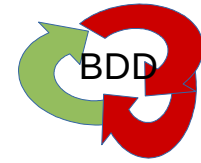
```
    suggestions.addSuggestionsAsBooks(books);
```

```
    return suggestions;
```

```
}
```

# Bibliothèque

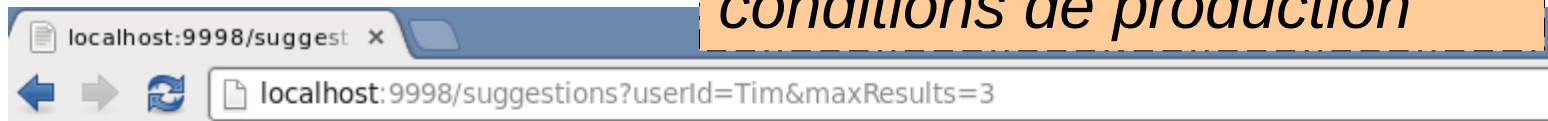
## Implémenter les scénarios



```
com.kelkoo.dojo.bdd.suggestions.bdd.fr.OnGoingBDDFrTest [Runner: JUnit 4] (0.286 s)
  Feature: Fournir des suggestions de livres (0.286 s)
    Scenario: les livres proposés sont populaires
      Given I utilisateur depuis le web service
      And les catégories depuis le web service
      And les livres depuis le web service http://my.library.com/search?categories=cat1,cat2
      And les livres depuis le web service http://my.library.com/user/Tim/books (0.088 s)
      When on appelle http://localhost:9998/suggestions?userId=Tim&maxResults=3 (0.00 s)
      Then le code http retourné est "200" (0.012 s)
      Then les suggestions sont (0.166 s)
```

*Premier scénario  
implémenté !*

*Le code est activé dans les  
conditions de production*

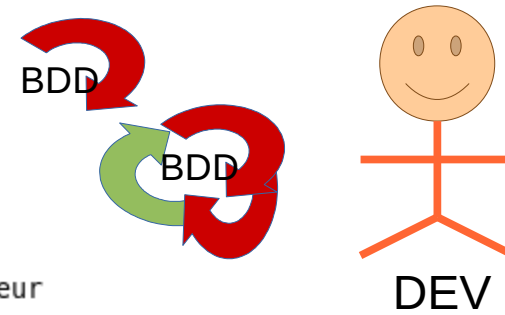


This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<?xml version="1.0" encoding="UTF-8"?>
<suggestions>
  <suggestions bookId="b11" bookTitle="Colorier les poules" categoryId="cat1"/>
  <suggestions bookId="b21" bookTitle="Comptines de la ferme" categoryId="cat2"/>
  <suggestions bookId="b31" bookTitle="Histoires de la mer" categoryId="cat3"/>
</suggestions>
```

# Bibliothèque

## Implémenter les scénarios



@level\_1\_specification @nominal\_case @ongoing

**Scenario:** les suggestions proposées sont populaires, disponibles et adaptées à l'âge de l'utilisateur

**Given** l'utilisateur "Tim"

**And** il a "4" ans

**And** les catégories populaires pour cet âge sont

categoryId	categoryName
cat1	Coloriage
cat2	Comptines
cat3	Histoires pour le dodo

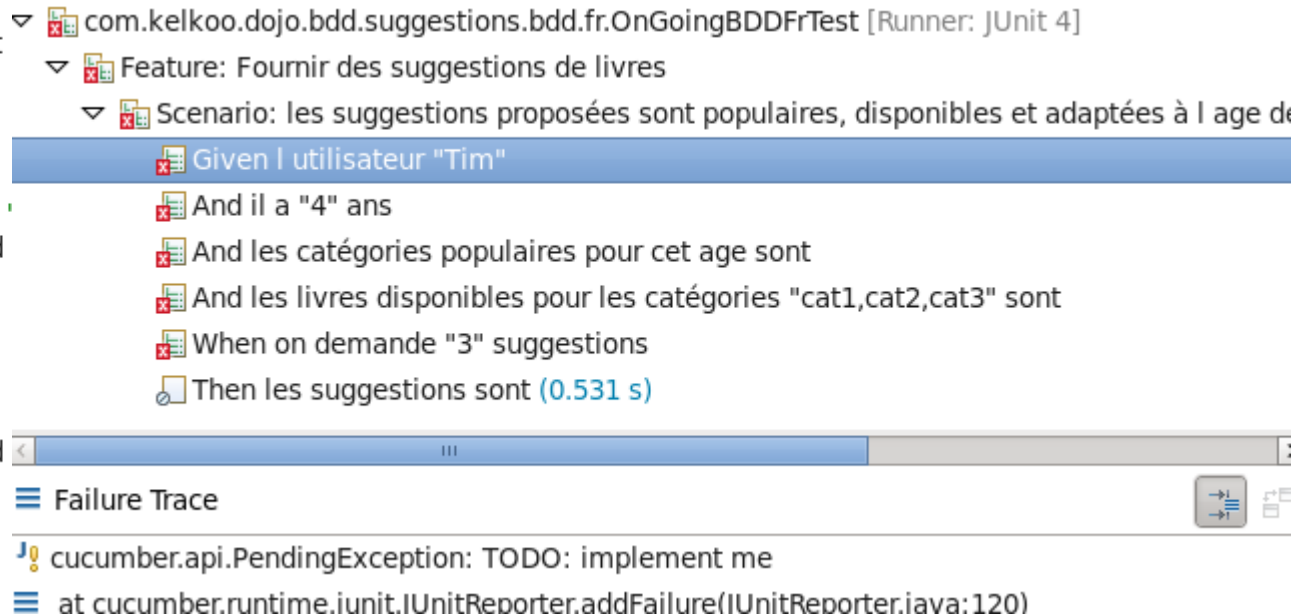
**And** les livres disponibles pour les catégories

bookId	bookTitle	categoryId
b11	Colorier les poules	cat1
b21	Comptines de la ferme	cat2
b31	Histoires de la mer	cat3

**When** on demande "3" suggestions

**Then** les suggestions sont

bookId	bookTitle	categoryId
b11	Colorier les poules	cat1
b21	Comptines de la ferme	cat2
b31	Histoires de la mer	cat3



*Réutilisation de phrase exécutable d'un niveau d'abstraction inférieur*

```
@Given("^l utilisateur \"([^\"]*)\"$")
```

```
public void given_the_user(String userId) throws Throwable {  
    given_the_user_from_user_ws( userId );  
}
```

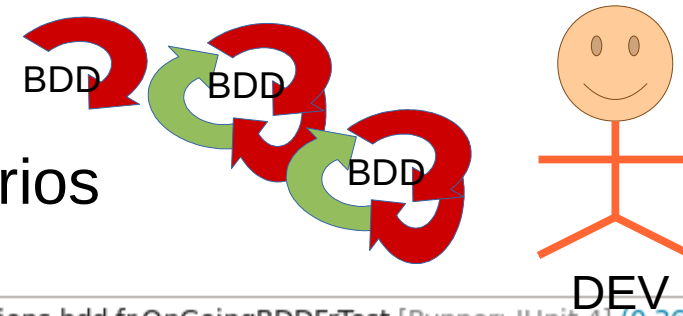
```
@Given("^les catégories populaires pour cet âge sont$")
```

```
public void given_the_popular_categories_for_this_age_are(List<Category> popularCategoriesGivenAgeUser) throws Throwable {  
    Boolean isPopular = true;  
    given_the_categories_from_categories_ws(isPopular, user.getAge(), popularCategoriesGivenAgeUser);  
}
```



# Bibliothèque

## Implémenter les scénarios



@level\_1\_specification @nominal\_case @ongoing

Scenario: limiter le nombre de suggestions

Given l utilisateur "Tim"

And il a "4" ans

And les catégories populaires pour cet age sont

categoryId	categoryName
cat1	Coloriage
cat2	Comptines
cat3	Histoires pour le dodo

And les livres disponibles pour les catégories "cat1,cat

bookId	bookTitle	categoryId
b11	Colorier les poules	cat1
b21	Comptines de la ferme	cat2
b31	Histoires de la mer	cat3

When on demande "2" suggestions

Then les suggestions sont

bookId	bookTitle	cate
b11	Colorier les poules	cat1
b21	Comptines de la ferme	cat2

- com.kelkoo.dojo.bdd.suggestions.bdd.fr.OnGoingBDDFrTest [Runner: JUnit 4] (0.36 s)
  - Feature: Fournir des suggestions de livres (0.364 s)
    - Scenario: les suggestions proposées sont populaires, disponibles et adaptées
      - Given l utilisateur "Tim" (0.002 s)
      - And il a "4" ans (0.011 s)
      - And les catégories populaires pour cet age sont (0.004 s)
      - And les livres disponibles pour les catégories "cat1,cat2,cat3" sont (0.085 s)
      - When on demande "3" suggestions (0.015 s)
      - Then les suggestions sont (0.007 s)
    - Scenario: limiter le nombre de suggestions (0.016 s)
      - Given l utilisateur "Tim" (0.001 s)
      - And il a "4" ans (0.001 s)
      - And les catégories populaires pour cet age sont (0.002 s)
      - And les livres disponibles pour les catégories "cat1,cat2,cat3" sont (0.004 s)
      - When on demande "2" suggestions (0.007 s)
      - Then les suggestions sont (0.001 s)

*Réutilisation de phrase exécutable*

### Failure Trace

java.lang.AssertionError:

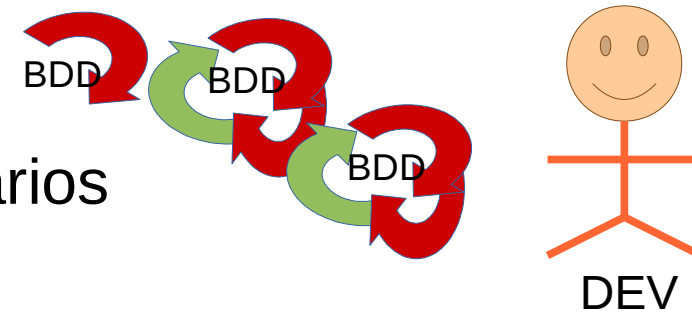
Expected: <2>

but: was <3>

at org.hamcrest.MatcherAssert.assertThat(MatcherAssert.java:20)

# Bibliothèque

## Implémenter les scénarios



```
@level_1_specification @nominal_case @ongoing
```

```
Scenario: limiter le nombre de suggestions
```

```
Given l utilisateur "Tim"
```

```
And il a "4" ans
```

```
And "3" livres sont disponibles pour les catégories populaires pour cet age
```

```
When on demande "2" suggestions
```

```
Then "2" suggestions sont proposées parmi les livres précédents
```

- com.kelkoo.dojo.bdd.suggestions.bdd.fr.OnGoingBDDFrTest [Runner: JUnit 4] (0.009 s)
  - Feature: Fournir des suggestions de livres (0.009 s)
    - Scenario: limiter le nombre de suggestions (0.006 s)
      - Given l utilisateur "Tim" (0.003 s)
      - And il a "4" ans (0.003 s)
      - And "3" livres sont disponibles pour les catégories populaires pour cet age
      - When on demande "2" suggestions (0.001 s)
      - Then "2" suggestions sont proposées parmi les livres précédents

Failure Trace

cucumber.api.PendingException: TODO: implement me

at cucumber.runtime.iunit.IUnitReporter.addFailure(IUnitReporter.java:1

*Générer des données pour rendre le scenario plus lisible*

```
@Given("^\"([^\"]*)\" livres sont disponibles pour les catégories populaires pour cet age$")
```

```
public void given_available_books_for_the_popular_categories_of_the_user_age(int nbBooks) throws Throwable {  
    given_the_popular_categories_for_this_age_are(asList( new Category("cat1","category1"), new Category("cat2","category2") ));  
    given_the_search_results_for_categories_are("cat1,cat2",  
                                                asList( new Book("b11","book11","cat1" ),  
                                                         new Book("b21","book21","cat2" ),  
                                                         new Book("b31","book31","cat3" )));  
}
```

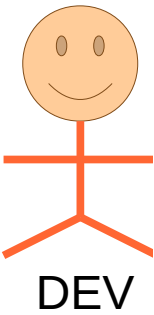
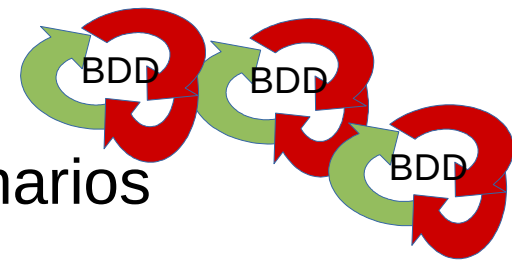
```
@Then("^\"([^\"]*)\" suggestions sont proposées parmi les livres précédents$")
```

```
public void then_suggestions_sont_proposées_parmi_les_livres_précédents(int nbSuggestions) throws Throwable {  
    then_the_suggestions_are(searchResult.subList(0, nbSuggestions) );  
}
```



# Bibliothèque

Implémenter les scénarios



Runs: 55/55

Errors: 0

Failures: 0

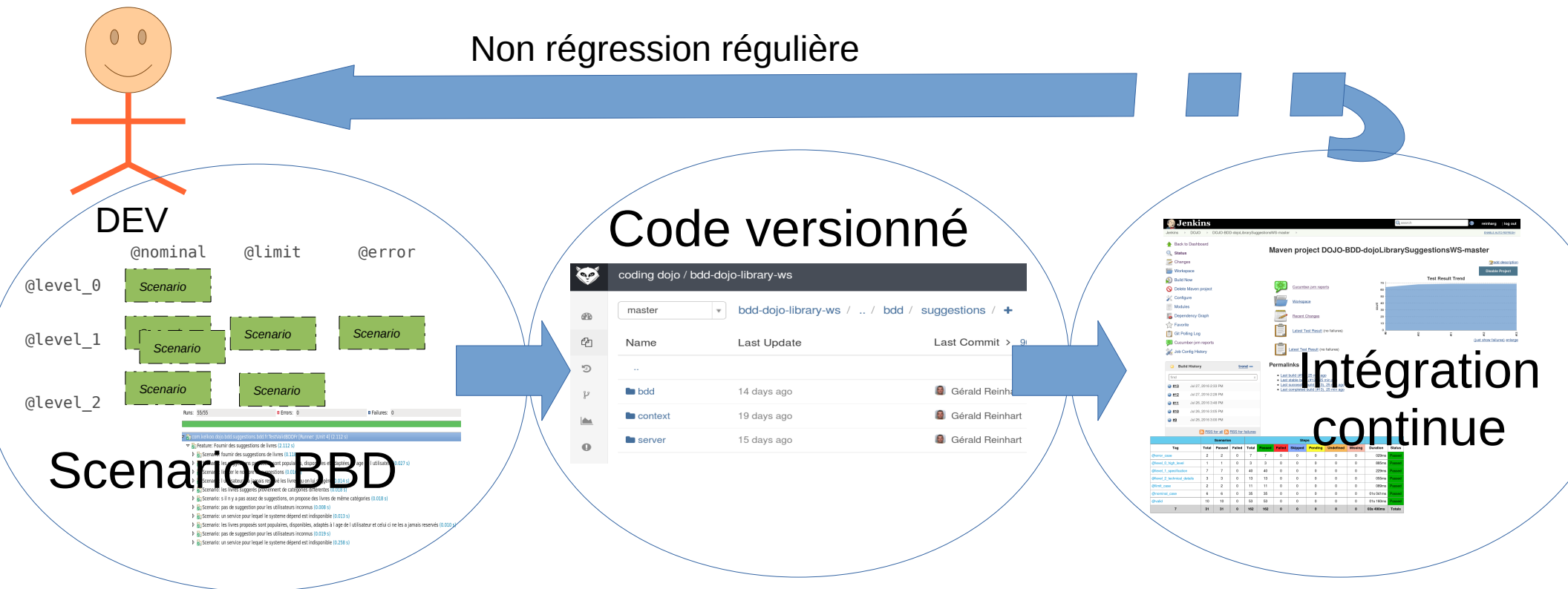
com.kelkoo.dojo.bdd.suggestions.bdd.fr.TestValidBDDFr [Runner: JUnit 4] (2.112 s)

Feature: Fournir des suggestions de livres (2.112 s)

- Scenario: fournir des suggestions de livres (0.118 s)
- Scenario: les suggestions proposées sont populaires, disponibles et adaptées à l'âge de l'utilisateur (0.027 s)
- Scenario: limiter le nombre de suggestions (0.016 s)
- Scenario: l'utilisateur n'a jamais réservé les livres qu'on lui suggère (0.014 s)
- Scenario: les livres suggérés proviennent de catégories différentes (0.018 s)
- Scenario: s'il n'y a pas assez de suggestions, on propose des livres de même catégorie (0.018 s)
- Scenario: pas de suggestion pour les utilisateurs inconnus (0.008 s)
- Scenario: un service pour lequel le système dépend est indisponible (0.013 s)
- Scenario: les livres proposés sont populaires, disponibles, adaptés à l'âge de l'utilisateur et celui-ci ne les a jamais réservés (0.010 s)
- Scenario: pas de suggestion pour les utilisateurs inconnus (0.019 s)
- Scenario: un service pour lequel le système dépend est indisponible (0.258 s)

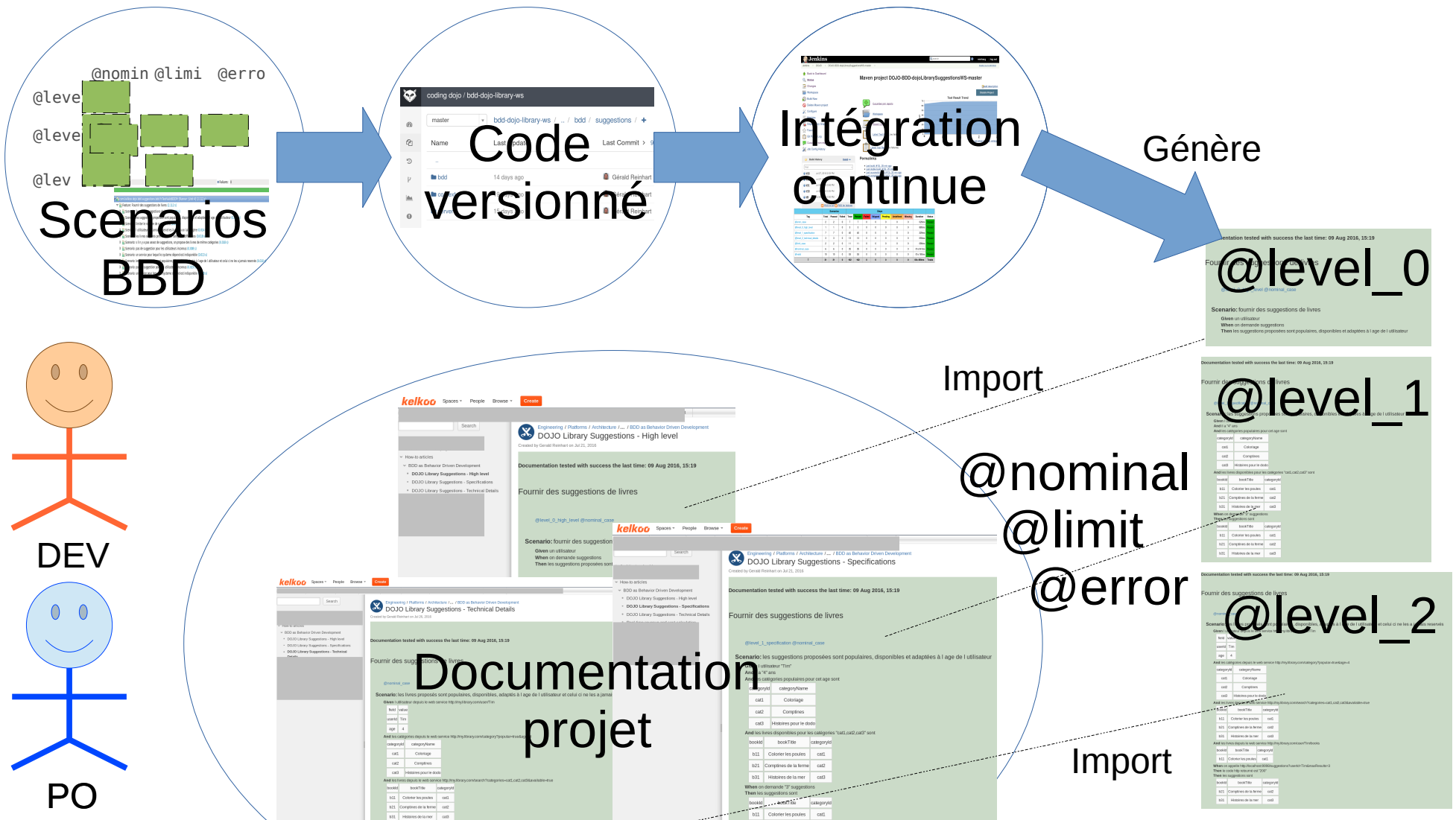
*Fin du cycle d'implémentation*

# Bibliothèque Tests de non régression



# Bibliothèque

Exposer la documentation générée



*Inclusion de la documentation générée dans la documentation projet en fonction du niveau d'abstraction et du type de scenario*

# Bibliothèque


## Exposer la documentation générée

**kelkoo** Spaces ▾ People Browse ▾ **Create**

▼ How-to articles

▼ BDD as Behavior Driven Development

- DOJO Library Suggestions - High level
- **DOJO Library Suggestions - Specifications**
- DOJO Library Suggestions - Technical Details

 Engineering / Platforms / Architecture / ... / BDD as Behavior Driven Development

**DOJO Library Suggestions - Specifications**

Created by Gerald Reinhart on Jul 21, 2016

**Documentation tested with success the last time: 09 Aug 2016, 15:19**

### Fournir des suggestions de livres

[@level\\_1\\_specification @nominal\\_case](#)

**Scenario:** les suggestions proposées sont populaires, disponibles et adaptées à l'âge de l'utilisateur

**Given** l'utilisateur "Tim"

**And** il a "4" ans

**And** les catégories populaires pour cet âge sont

categoryId	categoryName
cat1	Coloriage
cat2	Comptines
cat3	Histoires pour le dodo

**And** les livres disponibles pour les catégories "cat1,cat2,cat3" sont

bookId	bookTitle	categoryId
b11	Colorier les poules	cat1
b21	Comptines de la ferme	cat2
b31	Histoires de la mer	cat3

**When** on demande "3" suggestions

**Then** les suggestions sont

bookId	bookTitle	categoryId
b11	Colorier les poules	cat1

52

gerald.reinhart@kelkoo.com

# Conclusion

- Spécification par l'exemple
  - collaboration étroite DEV / PO est nécessaire
  - utiliser des exemples permet d'ouvrir la discussion et de trouver de nombreux cas
  - permet une boucle de rétroaction très rapide
- Tests fonctionnels
  - tests stables et rapides
  - le développeur est guidé, le code est tiré par les tests
  - le code doit être flexible pour mocker les interactions extérieures
- Documentation exécutable
  - issue du code, la documentation est à jour toujours
  - documentation exhaustive

# Conclusion

- Équipe plus soudée autour du projet
  - même niveau de compréhension pour tout le monde
  - les scénarios constituent un contrat clair
- Confiance
- Vélocité
  - Boucle de rétroaction très courte pour le PO
  - En cas de changement d'orientation produit la modification des tests et du code est plus rapide

# Conclusion

- Facteur d'échec
  - DEV ou PO non impliqués
  - BDD appliqué en cours de projet, doit être fait en premier

# Conclusion

Essayez !  
(ou réessayez !)

Questions ?