Propulsez votre architecture grâce au TDD et aux Mocks

ELAPSE technologies

FÉLIX-ANTOINE BOURBONNAIS

ING.JR, PSM, M.SC.

Agile Montréal

12 mars 2014

Image de Eyesplash
http://commons.wikimedia.org/wiki/File:Welkom_willkommen_Welcome_Bienvenue_Benvenuto.jpg

Félix-Antoine Bourbonnais

Ing. jr, PSM, M.Sc.

Formateur et **Coach** Agile

Tests automatisés

TDD

BDD et ATDD

Code **propre**

Qualité logicielle

Architecture Agile

Orientation objet

Orientation aspect

Design testable et émergeant

Scrum

Accompagnement d'équipes

Agent de changement

ELAPSE technologies

Formation – Accompagnement – Conseils

www.elapsetech.com

Contact

fbourbonnais@elapsetech.com

elapsetech.com/fab

@fbourbonnais

linkedin.com/in/fbourbonnais

Avertissement

Cette présentation s'adresse principalement à un **public ayant déjà expérimenté le TDD** mais aucune supervision n'est nécessaire pour en profiter…

Réchauffement…

Qui fait du TDD?

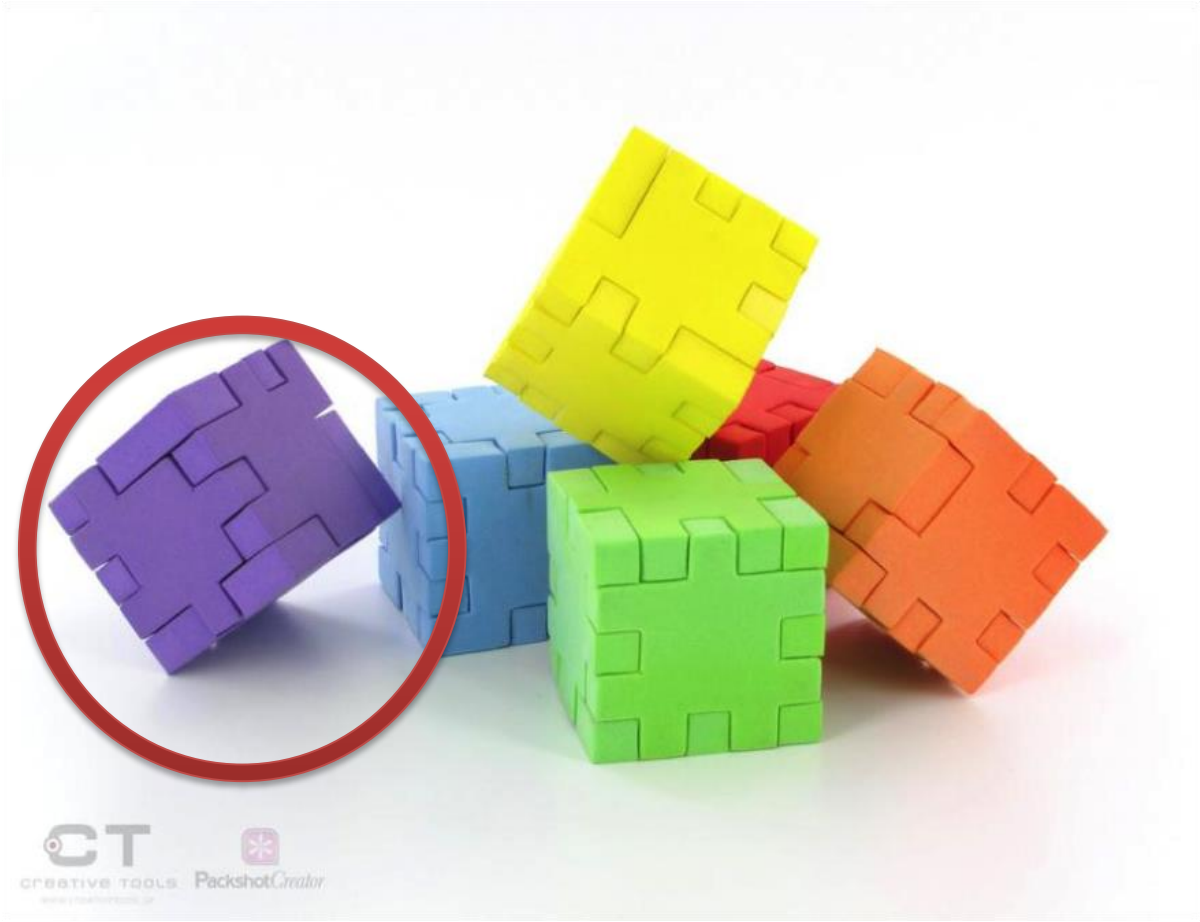
Qui utilise des Mocks?

Quelles sont vos attentes?

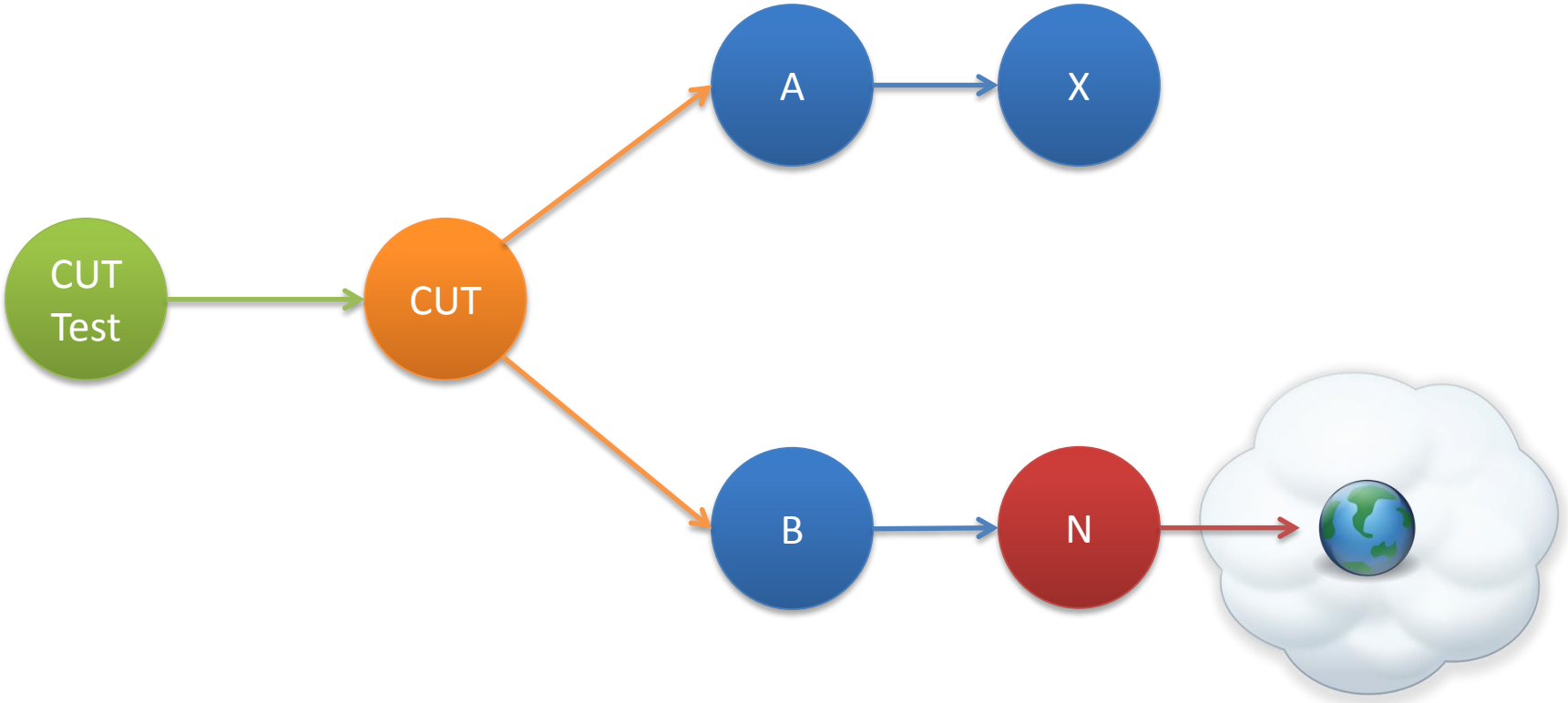
Rappel sur les

DOUBLURES ET MOCKS

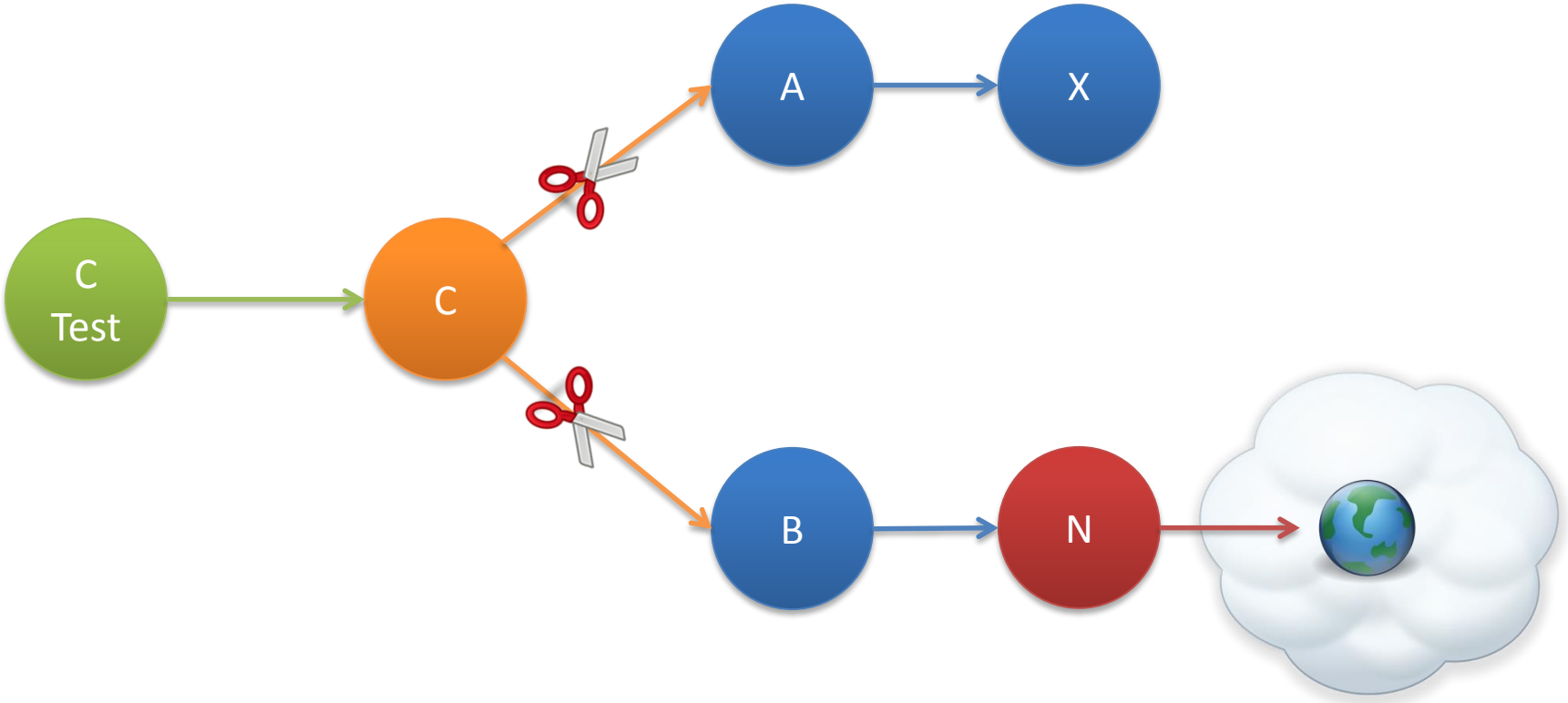
Rappel: les tests unitaires



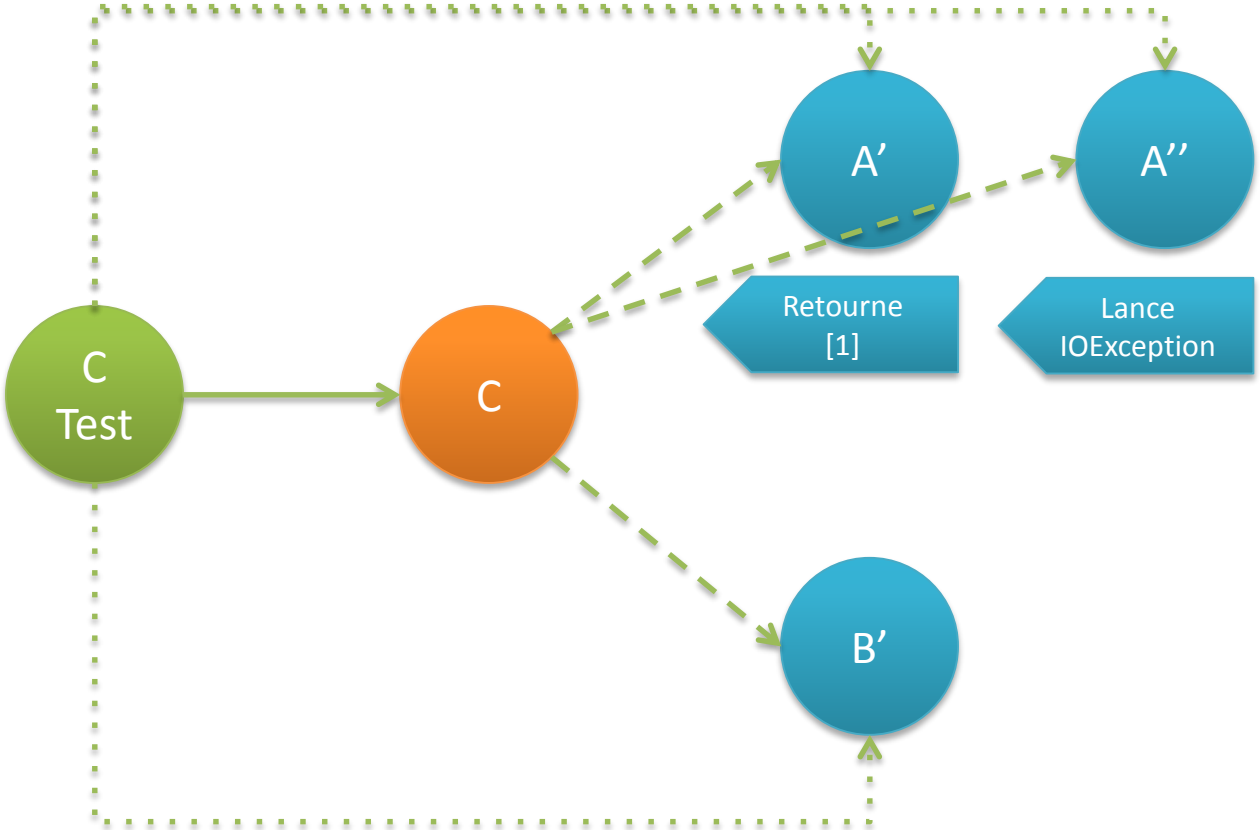
Rappel: les Mocks



Rappel: les mocks



Rappel: les mocks



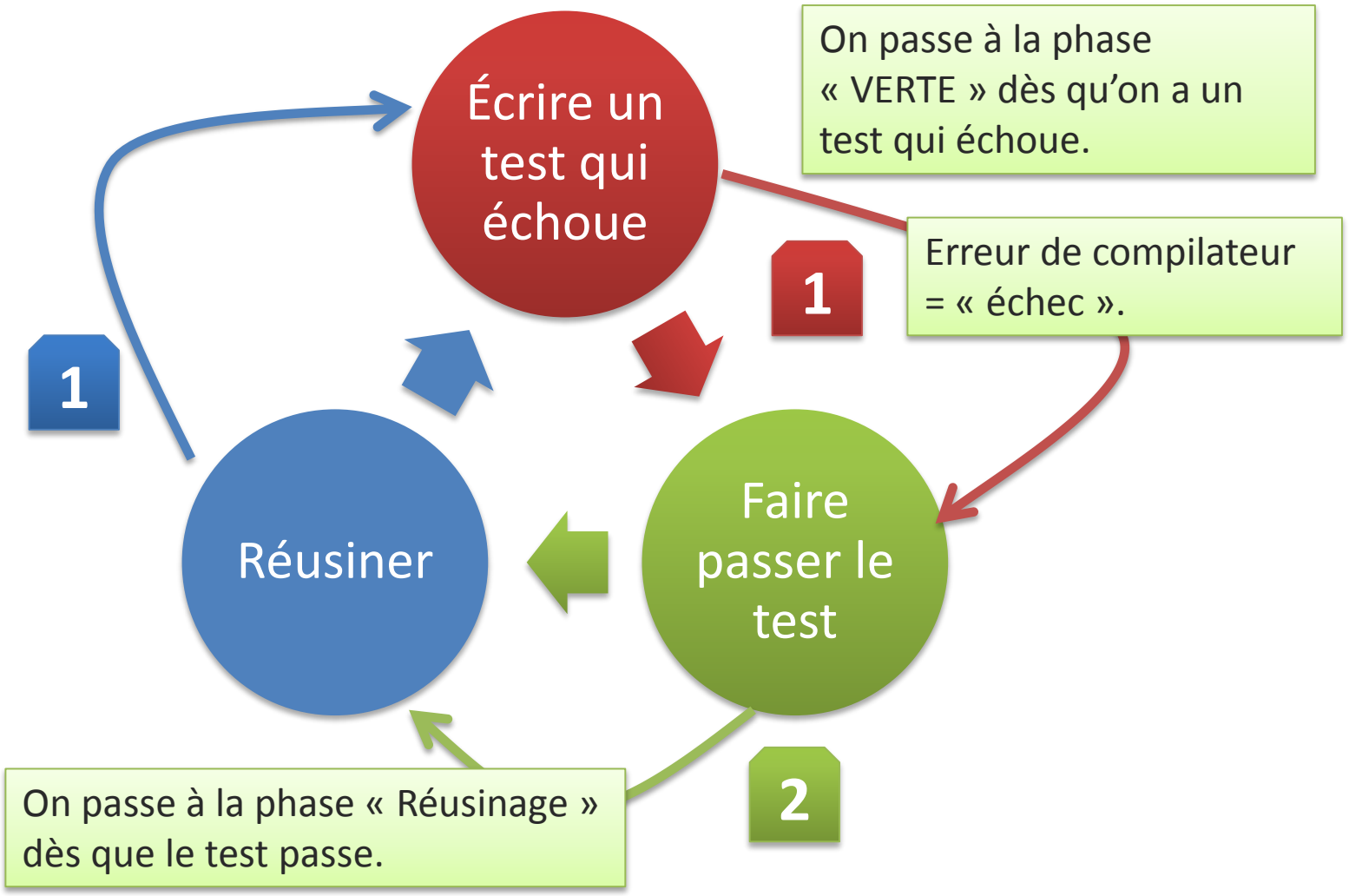
Rappels des fondements du

TDD

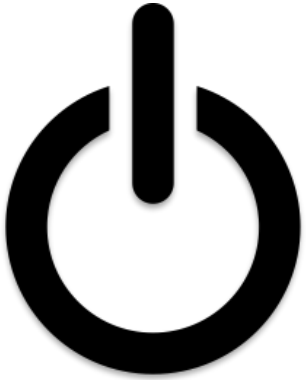
Technique ou discipline?

Le TDD est une discipline

Rappel: Le cycle du TDD



Shu-Ha-Ri

SHU	HA	RI
		
L'élève suit l'enseignement d'un maître	Il apprend de d'autres maîtres (écoles)	Il suit et a sa propre pratique…

Le design et le

TDD « CLASSIQUE »

Image: posterize / FreeDigitalPhotos.net

TDD classique

Centré sur l'**état**
et le **résultat final**

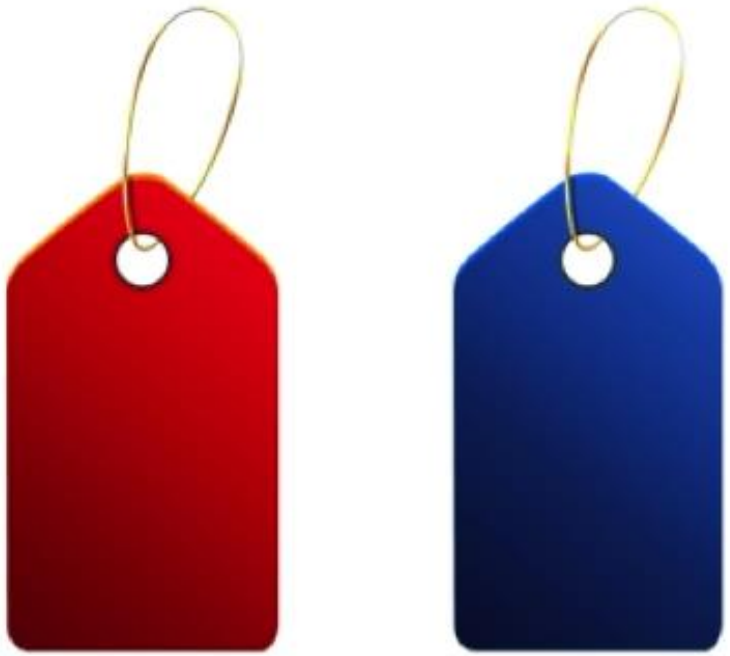
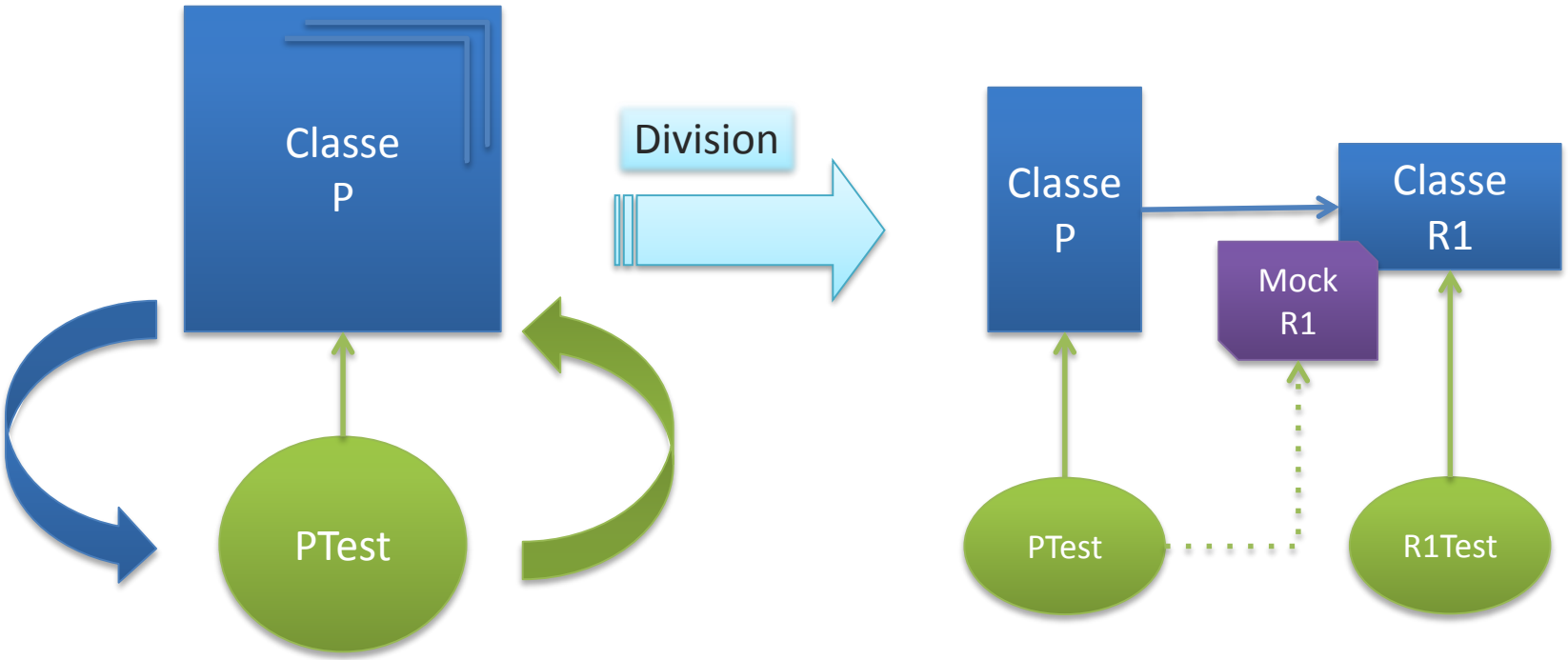


Image: nuchylee / FreeDigitalPhotos.net

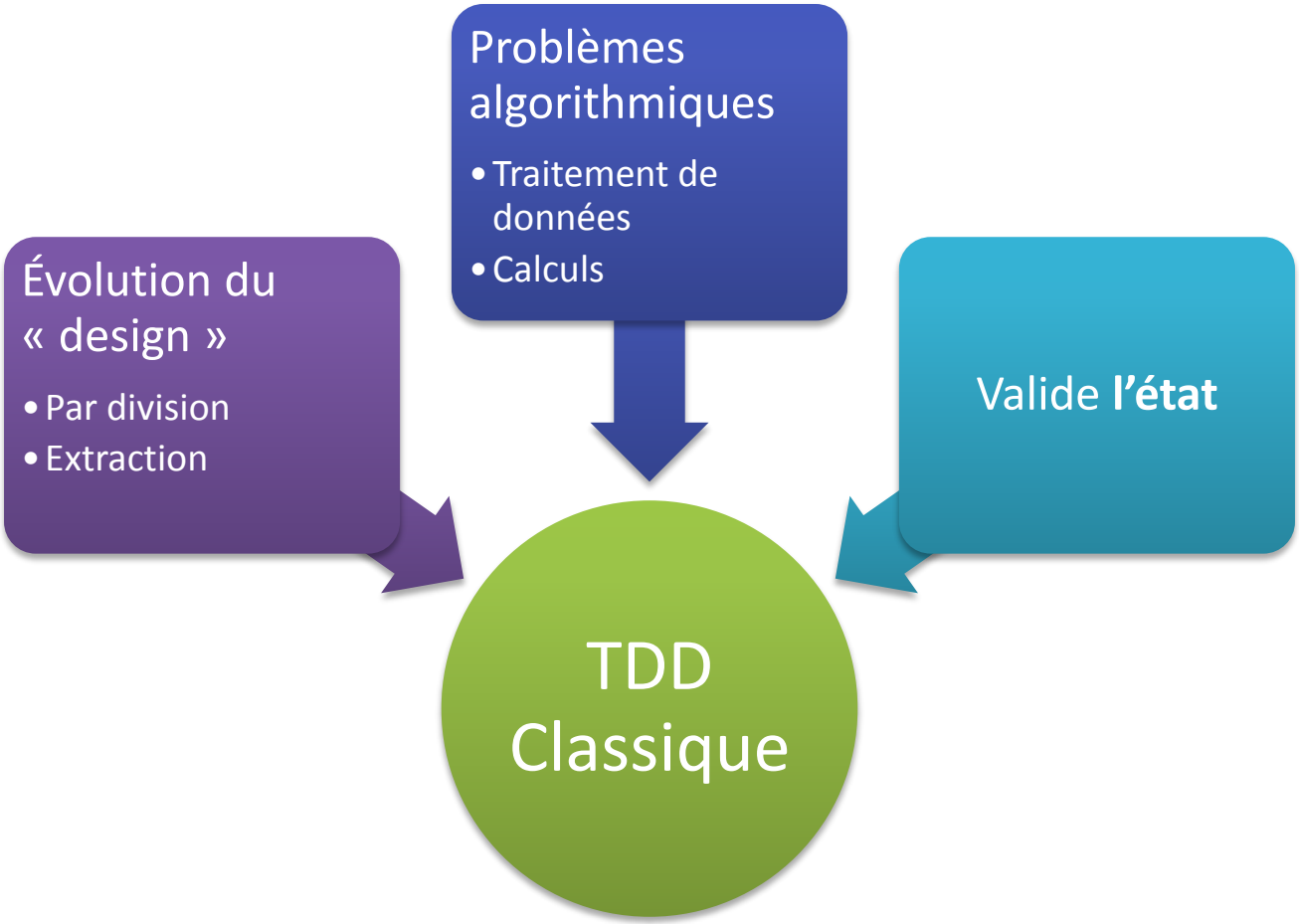
TDD classique

Extraction des types



TDD classique

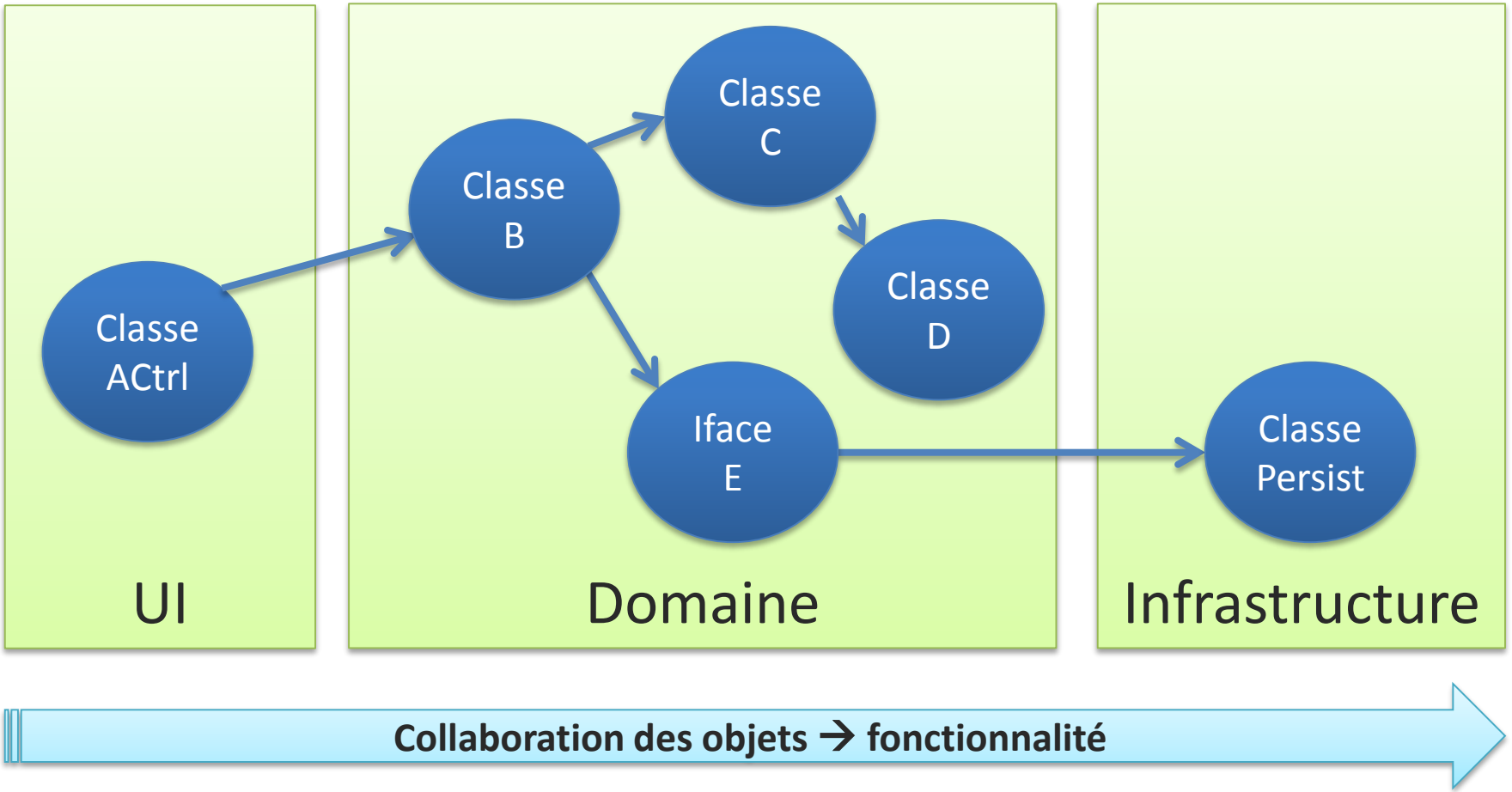
En résumé



Rappel de

L'ORIENTATION OBJET

Messages et collaboration



Le « Tell don’t Ask »

```
getAmi(joe)
    .getCorps()
    .getJambe(Orient.GAUCHE)
    .setPosition(
       sol + 5,
       centre + 20)
```

Hein !?!

Image: renjith krishnan et jscreationzs / FreeDigitalPhotos.net

Pourquoi le « Tell don’t ask »

Cacher le « comment »

Limiter l’effet des modifications

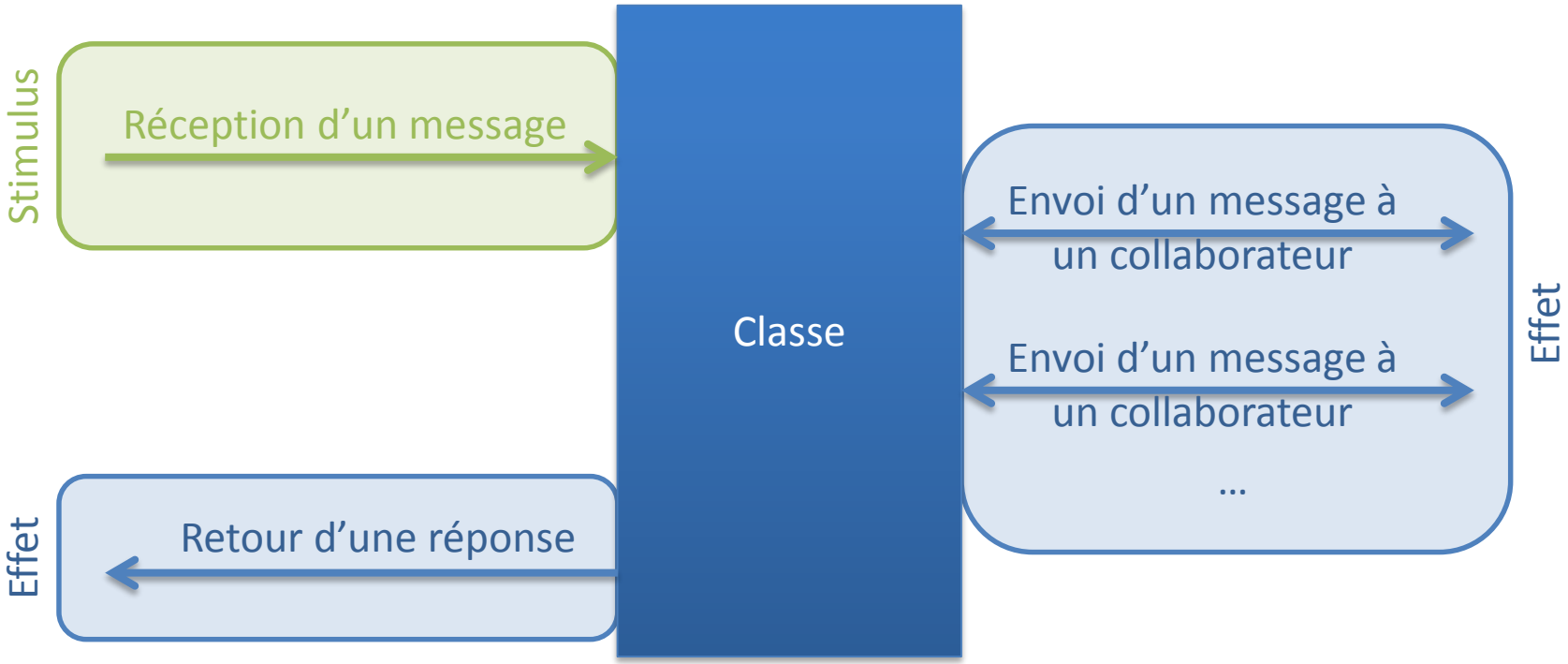
Limiter l’effet d’avalanche

Réduire la duplication

Laisser les objets « s’occuper de leurs oignons »

Éviter les « domaines anémiques »

Un objet est une boîte noire



Comment

PILOTER SON DESIGN AVEC DES MOCKS?

Le TDD « Mockiste »

Centré sur les **interactions** et **comportements**
entre les **objets** considérant leur **rôle**

Le TDD « Mockiste »

Utilise les **Mocks** comme pierre angulaire

TDD « Mockiste »

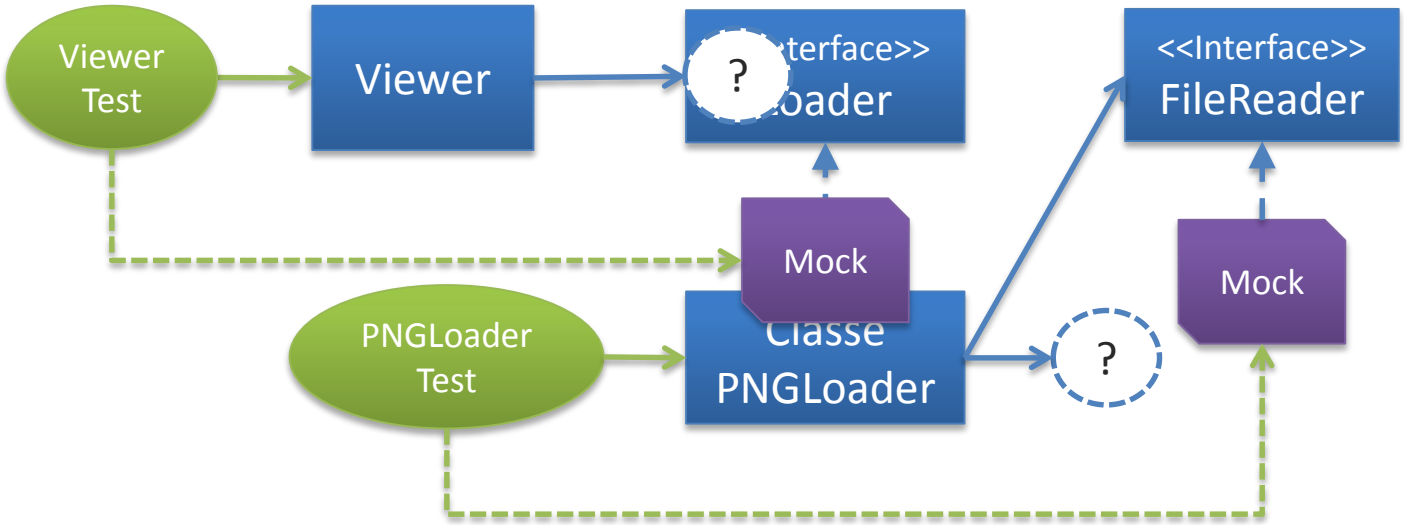
Extraction des types

Découverte des types par les **Mocks**

Définition de l'interface à partir des **besoins établis dans les autres tests**

TDD piloté par les Mocks

Identifier les rôles

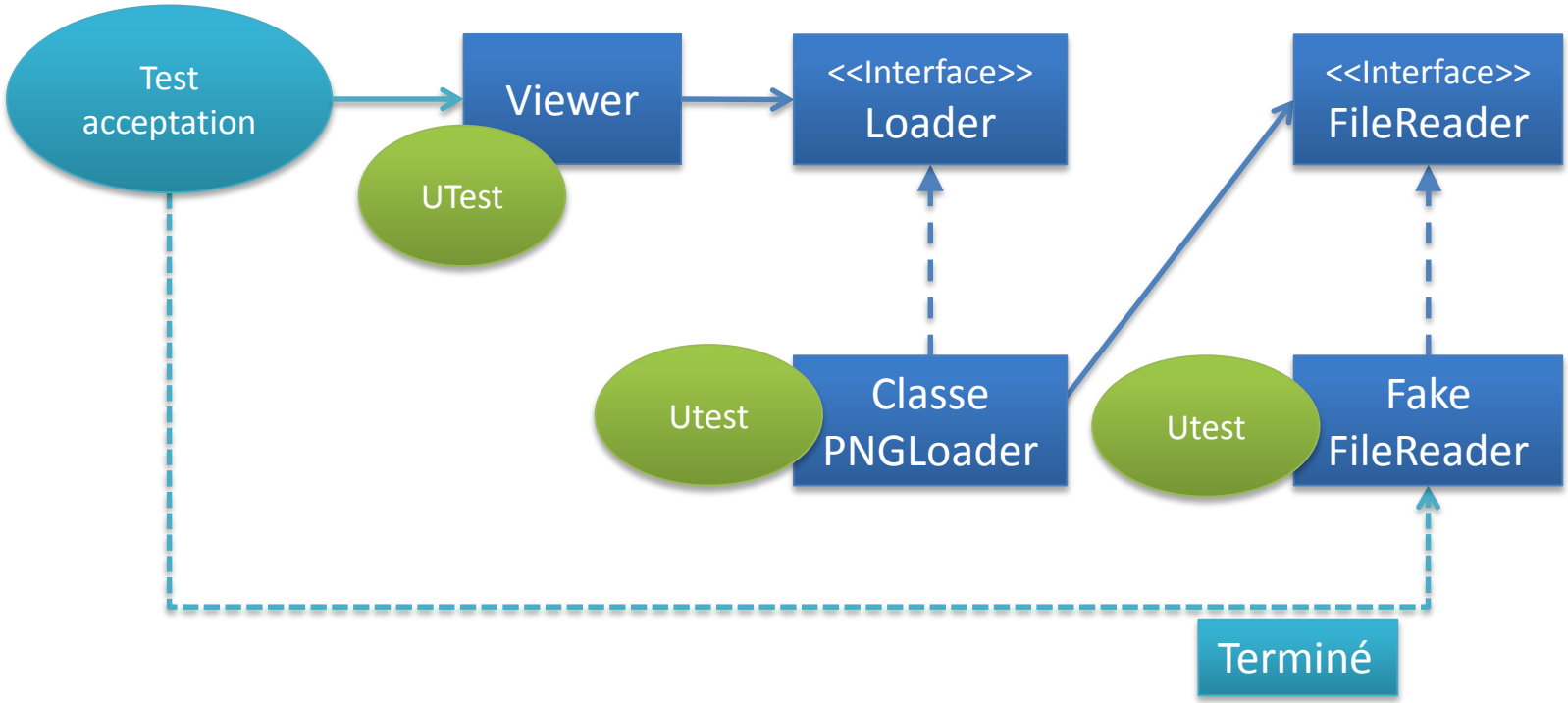


Découverte pas à pas

Tirer les types à partir de la demande

TDD piloté par les Mocks

Arriver à destination…

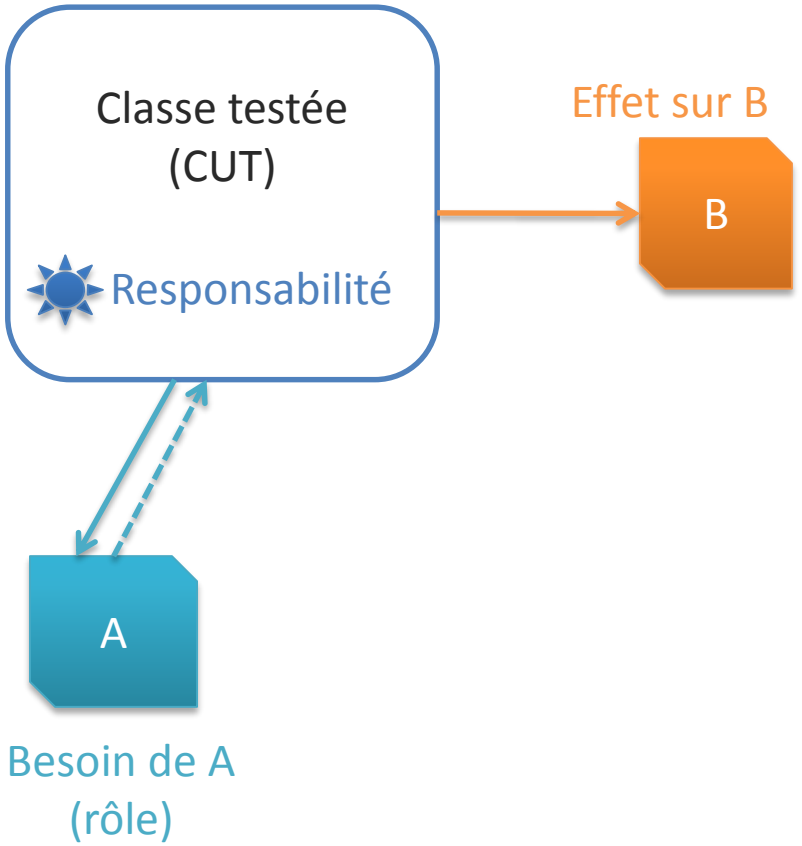


En résumé

Quelle est la **responsabilité** de l'objet testé ?

De quoi est-ce que l'objet a **besoin** pour réaliser son travail en terme de **rôles** ?

Quels **effets externes** aura ce comportement sur l'environnement immédiat?



Exemple

```
banque.acheter(carte, marchand, montant)
    --> carte.crediter(montant)
    --> marchand.debiter(montant)
```

Présentation de la

DÉMONSTRATION

Image: Stuart Miles / freedigitalphotos.net

Démonstration

Soumissions à une conférence

#1 Soumission d'une présentation

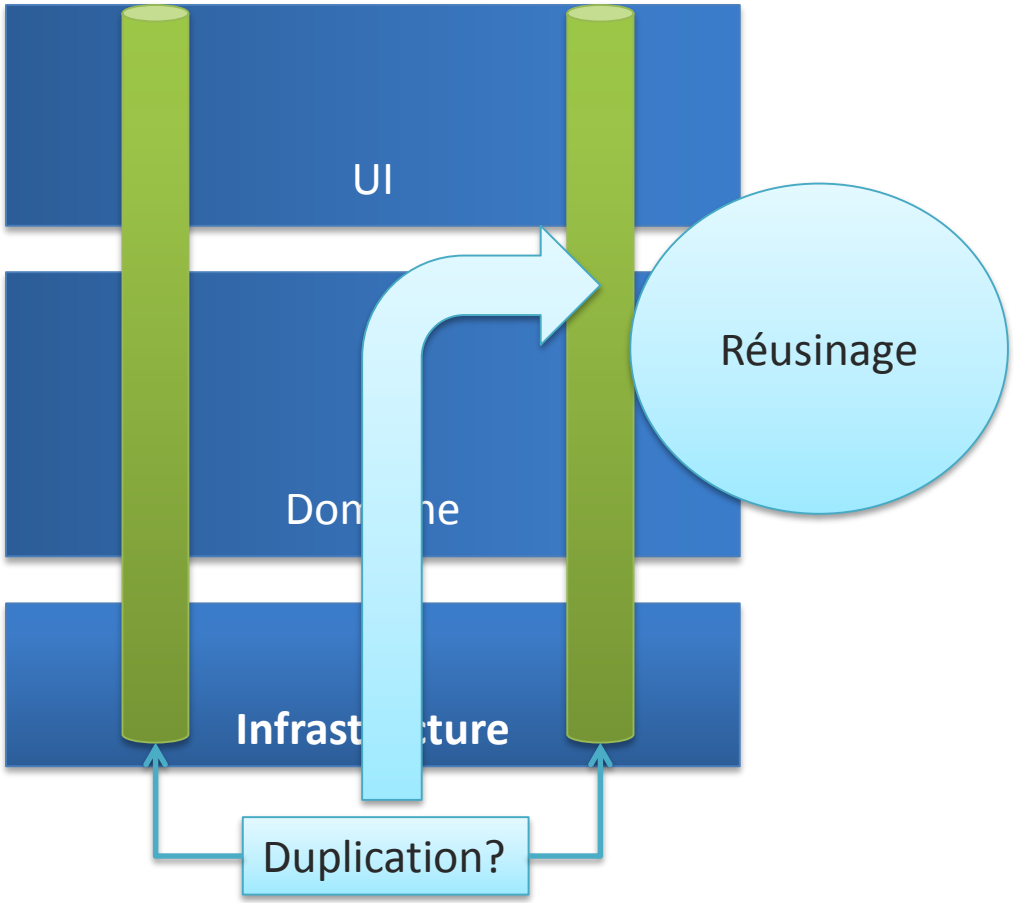
En tant que soumissionnaire
Je veux soumettre une présentation à une conférence
Afin qu'elle soit évaluée par le comité de sélection

Critères d'acceptation

- Il est possible de soumettre une présentation
- La présentation est accumulée en attendant d'être évaluée par le comité
- Le comité est avisé qu'une nouvelle présentation doit être évaluée

CONCLUSION et RÉSUMÉ

Approche « outside-in »



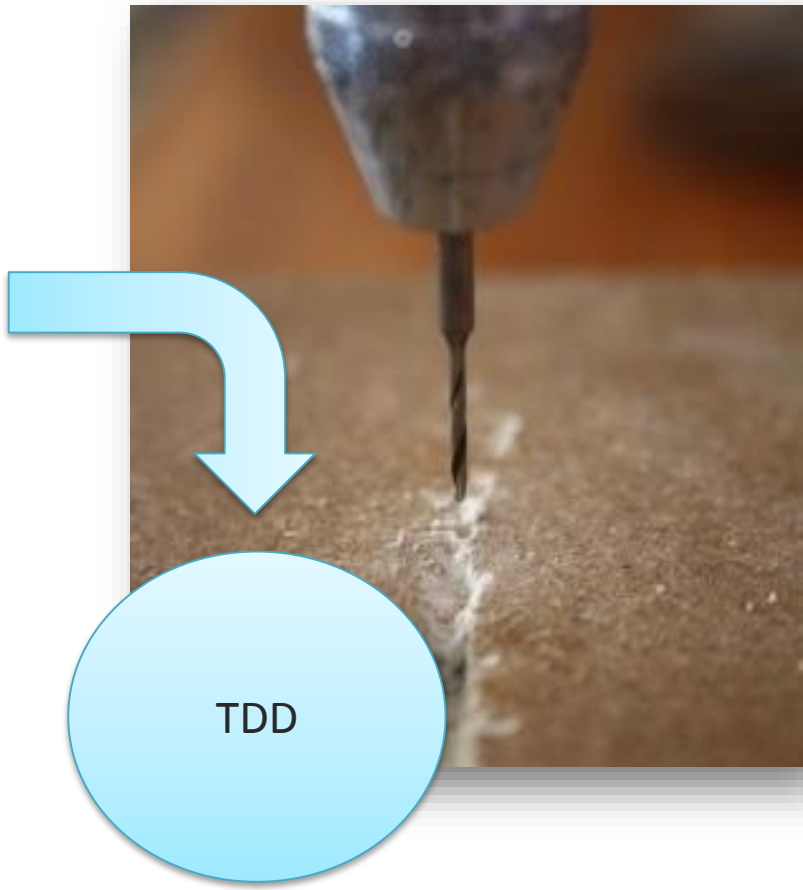
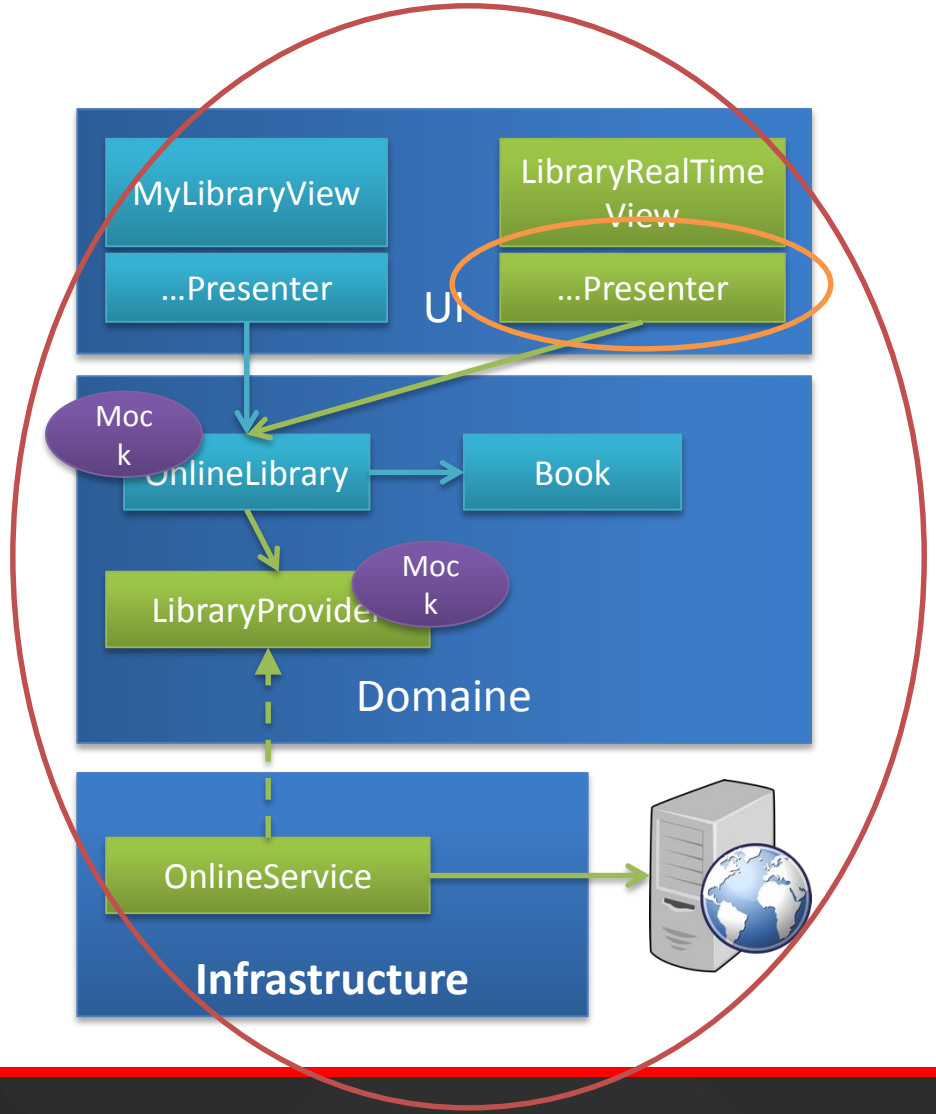


Image: Simon Howden / FreeDigitalPhotos.net

Piloter le design par les mocks



Composer à partir des interactions

Position « utilisateur »

Explorations successives (étape par étape)

Reporter les décisions d'implémentations

Explorer sans trop se compromettre

Image: Simon Howden / FreeDigitalPhotos.net

Avantages de l'approche mockiste

Favorise le « Tell don't ask »

Moins de « trains d'appels » (Demeter)

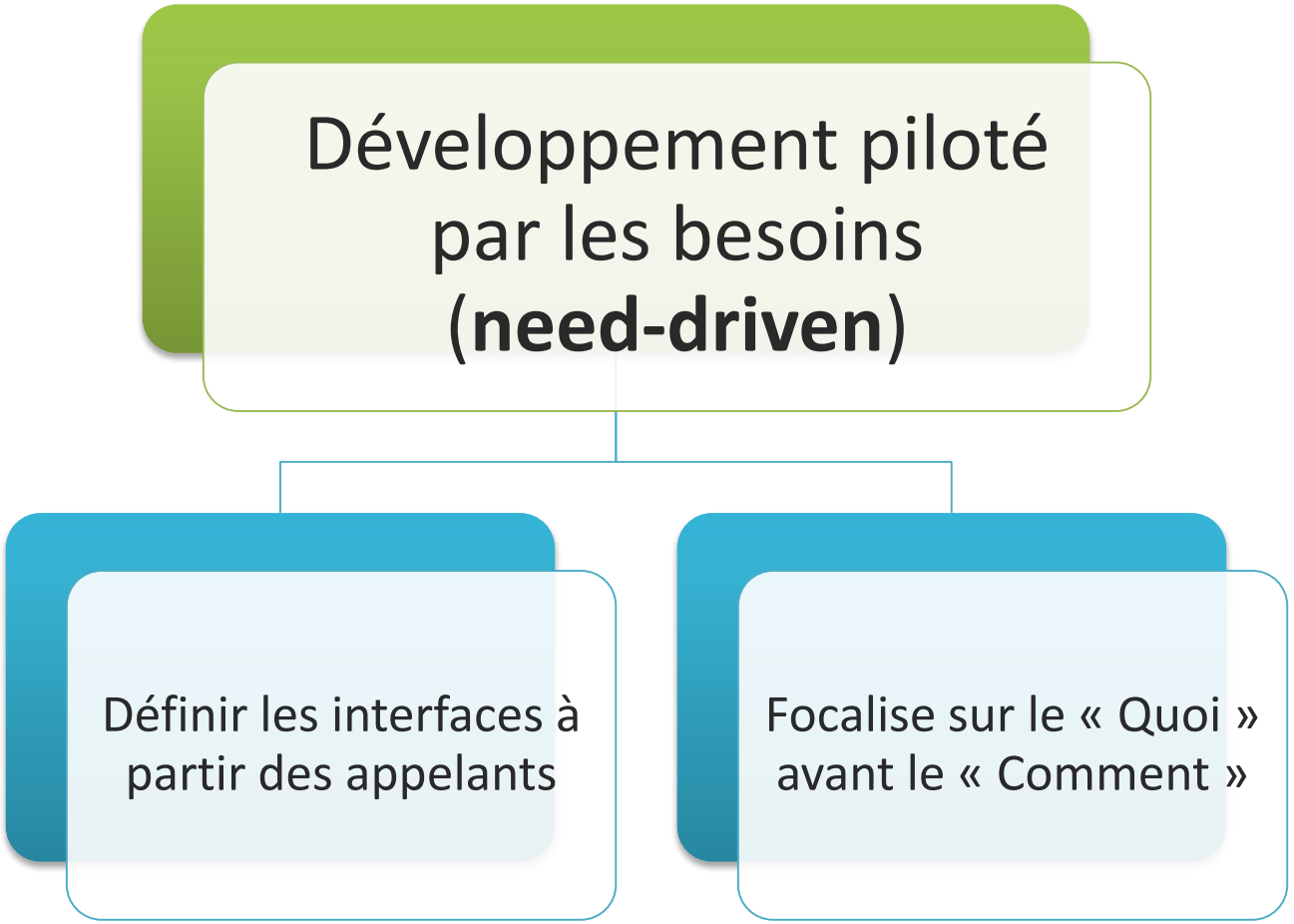
Retarde les décisions d'implémentations

Favorise un design testable

Requiert moins d'objets implémentés pour avoir une rétroaction

Classe fautive ciblée en cas d'échec

Avantages de l'approche mockiste



Ce que l'on obtient généralement

Hiérarchie mince

Design basé sur les rôles

Abstraction (rôles)

Nommage clair pour l'appelant

Possible meilleur respect des principes OO

Désavantages de l'approche mockiste

- Couplage du test avec la signature des collaborateurs
- Fragiliser les tests
- Fonctionne mal avec des problèmes algorithmiques
- Peut générer beaucoup de Mocks et d'interfaces
- Manquer de tests intégrés (pyramide)

La **bonne question**…

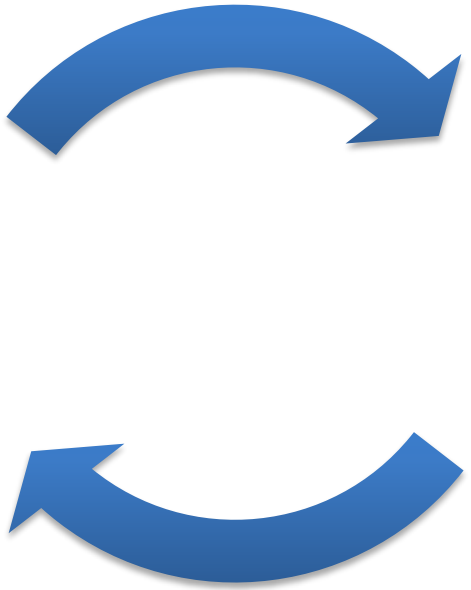
Que voulez-vous maximiser ?

Complémentarité

Cette école doit être **combinée**!

Alterner entre les techniques apporte généralement de bons résultats.

Choisir selon ce que l'on veut **découvrir**



Rappel: TDD et architecture

+
Code testable

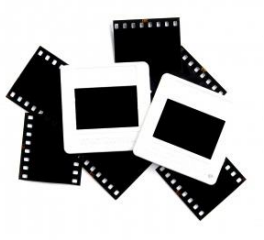
+ de
« design »
simple?

- de code
couplé

+ de code
réutilisable

+
d'architecture
émergeante

Pour aller plus loin…



Diapositives

http://developpementagile.com/architecture-mocks-tdd

Code source

https://github.com/fbourbonnais/propulsez-architecture-tdd-mocks

Nos **formations** sur le sujet

- Les **tests** en agilités
- **TDD**
- **TDD avancé**
- Concepts **OO avancés**
- **TDD** pour **gestionnaires**
- Introduction à **l'ATDD** et **BDD**

www.elapsetech.com/formation

Image de anamkkml/ FreeDigitalPhotos.net et github.com

Merci

Mon nom
Félix-Antoine Bourbonnais

Notre blogue
developpementagile.com

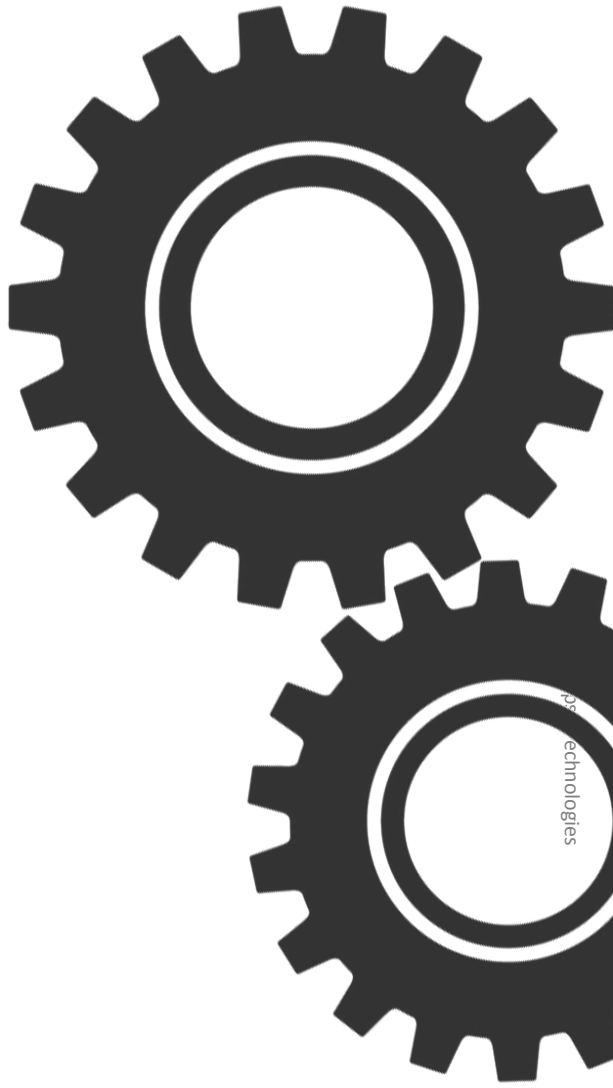
Notre site
elapsetech.com

Mon Twitter
@fbourbonnais

Mon LinkedIn
linkedin.com/in/fbourbonnais/fr

Elapse Technologies

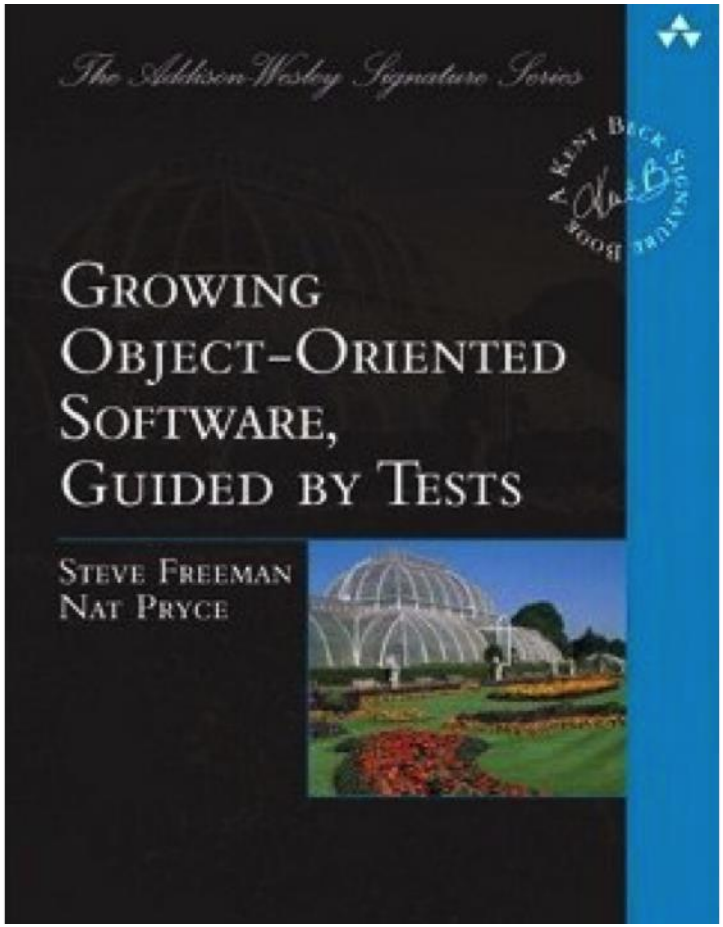
Votre allié en développement logiciel Agile



- **Formation**
- **Accompagnement** (coaching)
- Mentorat **virtuel**
- **Conseils** et **diagnostics**

- **Agilité** (Scrum, Lean, XP)
- Qualité et **tests** automatisés
- **Architecture** Agile
- **Pratiques** de développement

Références



Références

Steve Freeman, Tim Mackinnon, Nat Pryce, et Joe Walnes. *Mock roles, Not objects.* p. 236–246. OOPSLA ’04. Vancouver, BC, Canada, ACM, 2004.

Nat Pryce, et Steve Freeman, *InfoQ: Mock Roles Not Object States* . QCon London 2007 http://www.infoq.com/presentations/Mock-Objects-Nat-Pryce-Steve-Freeman

Martin Fowler, *Mocks Aren’t Stubs*, 2 janvier 2007. [Résumé des approches] http://martinfowler.com/articles/mocksArentStubs.html

Références

Steve Freeman, *Sustainable Test-Driven Development*. QCon San Francisco 2009.
http://www.infoq.com/presentations/Sustainable-Test-Driven-Development

Codemanship presents... Classic TDD vs. London School, 2011. [Critiqué]
http://www.youtube.com/watch?v=AUE155LISV4

Michael Feathers et Steve Freeman. *Michael Feathers and Steve Freeman on Design*, InfoQ at QCon San Francisco 2009
http://www.infoq.com/interviews/feathers-freeman-design

Discussion – *« Classic TDD or « London School » - any opinions/comments/elaboration on Jason Gorman's post? »* GOOS Mailinglist, 2011.
https://groups.google.com/d/topic/growing-object-oriented-software/dOmOIafFDcI/discussion