
SoftFluent
Les Bonnes PRATIQUES DU TEST LOGICIEL

SOMMAIRE

Qu’est-ce que le test logiciel ?

Pourquoi le test est-il un maillon crucial de l’ingénierie logicielle ?

Quels sont les différents types de tests ?

Qu’est-ce que la testabilité ?

Quels sont les scénarios, données, critères d’acceptation et pertinence pour chaque type de tests ?

Comment mettre le test au centre de la problématique projet ?

En pratique chez SoftFluent

Comment faire lorsque l’on a un logiciel existant ?

QU'EST-ce que le test logiciel ?

Le test est une discipline vaste qui permet de s'assurer de la qualité des logiciels avant leur mise sur le marché !

Toute conception quelle qu'elle soit nécessite une phase de test

La conception logicielle ne devrait pas déroger à cette règle

Pourtant, le test fait souvent l'objet d'un investissement insuffisant

QU'EST-ce que le test logiciel ?

Le test n'est pas une solution miracle!
C'est une recette mais elle n'est pas magique

Les tests augmentent fortement les chances de réussite d'un projet de développement

QU'EST-ce que le test logiciel ?

C'est une recette mais elle n'est pas magique, elle nécessite ….

… UNE ÉQUIPE

QU'EST-ce que le test logiciel ?

C'est une recette mais elle n'est pas magique, elle nécessite : une équipe, …

… DE LA RIGUEUR

QU'EST-ce que le test logiciel ?

C'est une recette mais elle n'est pas magique, elle nécessite : une équipe, de la rigueur et …

… DE L'EXPERTISE

QU'EST-ce que le test logiciel ?

Il est important qu'il intervienne dans le bon timing!

Plus les tests arrivent tard plus ils sont complexes à réaliser

Les anomalies sont plus faciles à comprendre et corriger sur un périmètre limité

Les tests sont tout aussi importants après la mise en production

Les jeux de tests doivent être réalisés tôt dans le cycle pour lever les erreurs de conception et faciliter le travail des équipes de développement

QU'EST-ce que le test logiciel ?

Au moins un tiers de l'effort de création logicielle!

Quelle que soit la méthode les tests ne doivent

pas être sous-estimés

Ils demandent de la rigueur, du temps, de la méthodologie

Le test est un vrai métier qui doit faire partie intégrante du projet de développement

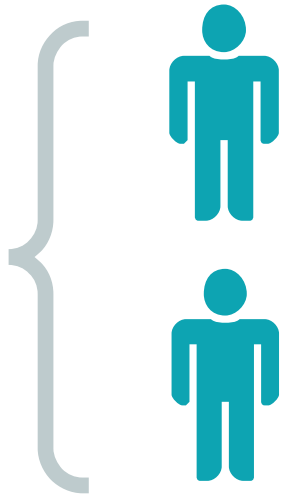
Pourquoi le test est-il un maillon crucial ?

Les tests sont garants de la qualité logicielle (extérieure, intérieure et future) !

Les différents types de tests permettent de garantir la qualité logicielle à condition de :

Bien dimensionner son équipe :

1 testeur



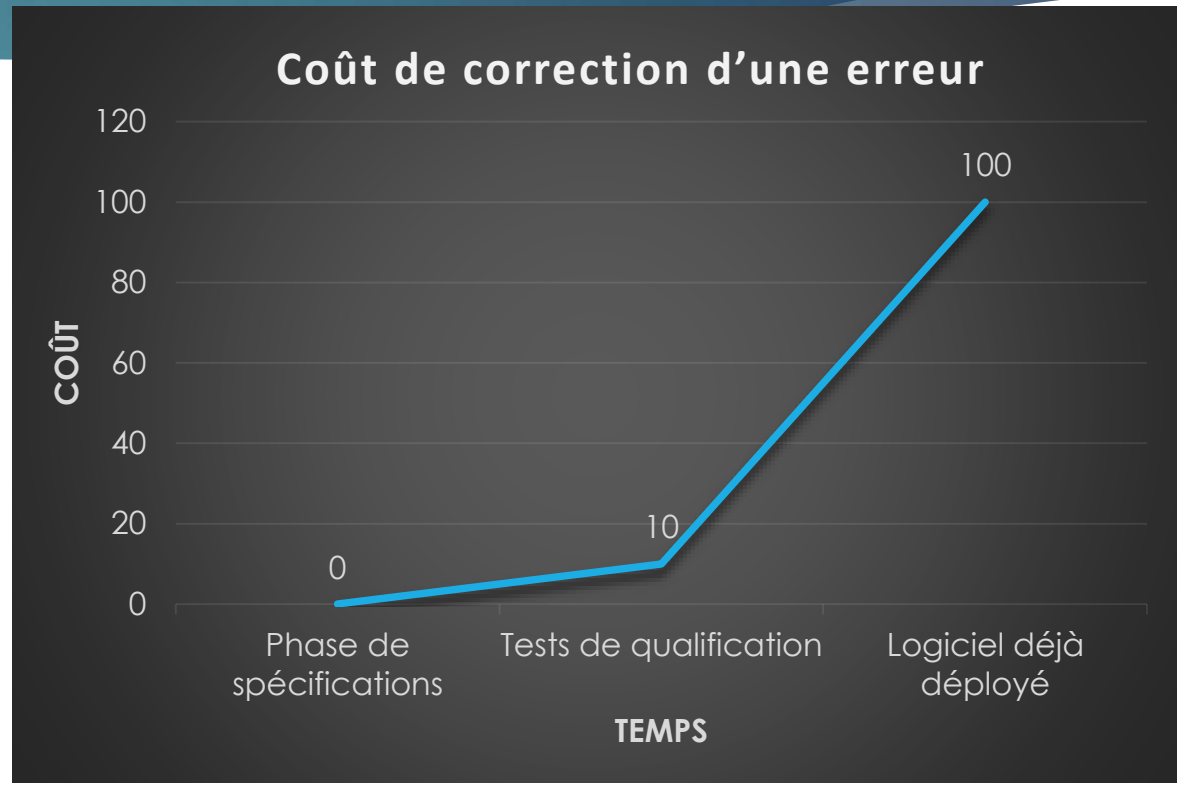
2 développeurs

Automatiser et/ou outiller les tests au maximum

Il faut de réels rôles de testeurs distincts des développeurs

Pourquoi le test est-il un maillon crucial ?

Le coût d'une anomalie est très différent suivant la phase de détection



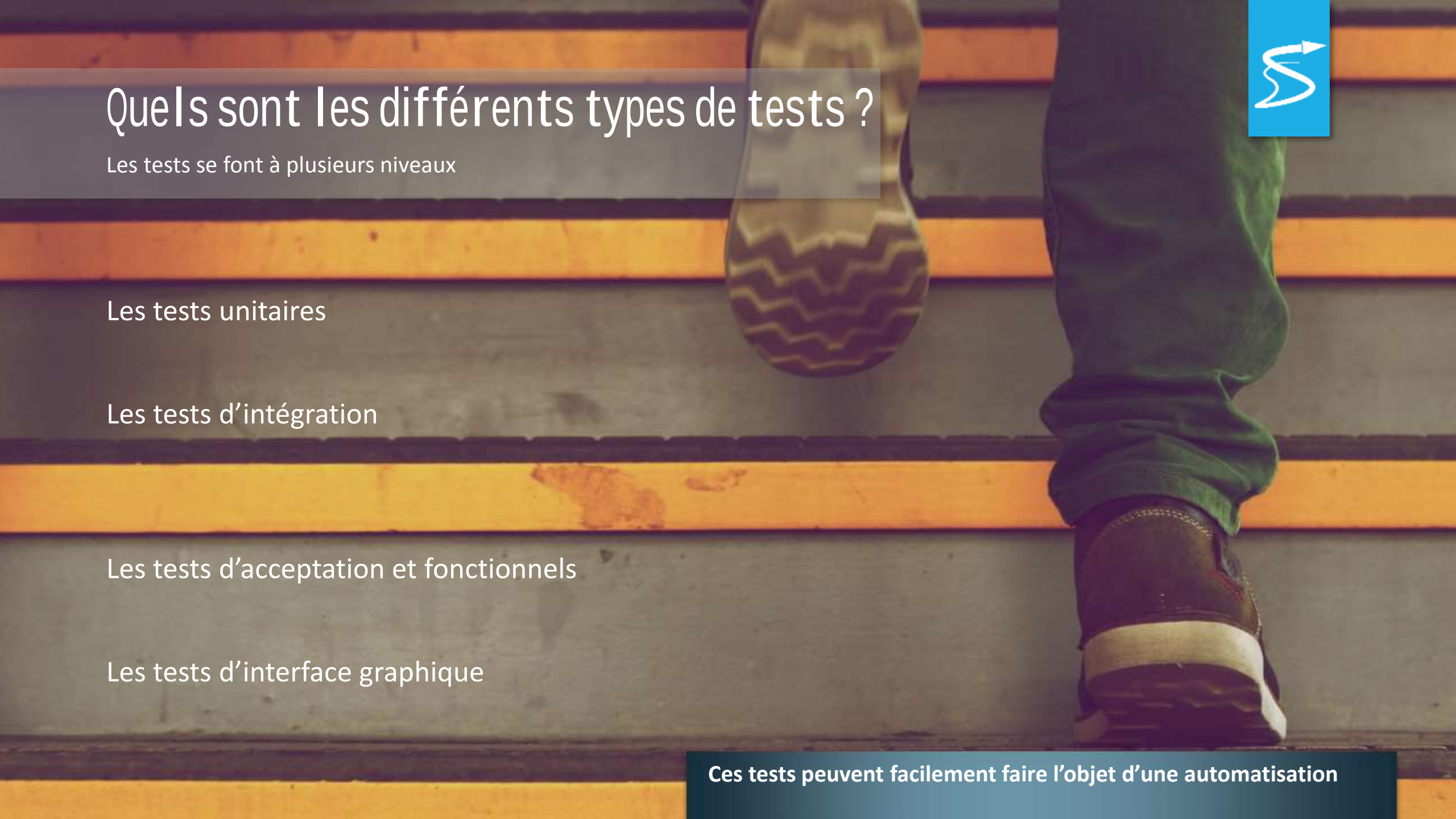
SI une erreur de conception est détectée lors :

De la phase de spécifications alors **coût = 1**

Des tests de qualification **coût = 10**

Si le logiciel est déjà déployé **coût = 100**

Le test permet la détection précoce des anomalies



Quels sont les différents types de tests ?

Les tests se font à plusieurs niveaux

Les tests unitaires

Les tests d'intégration

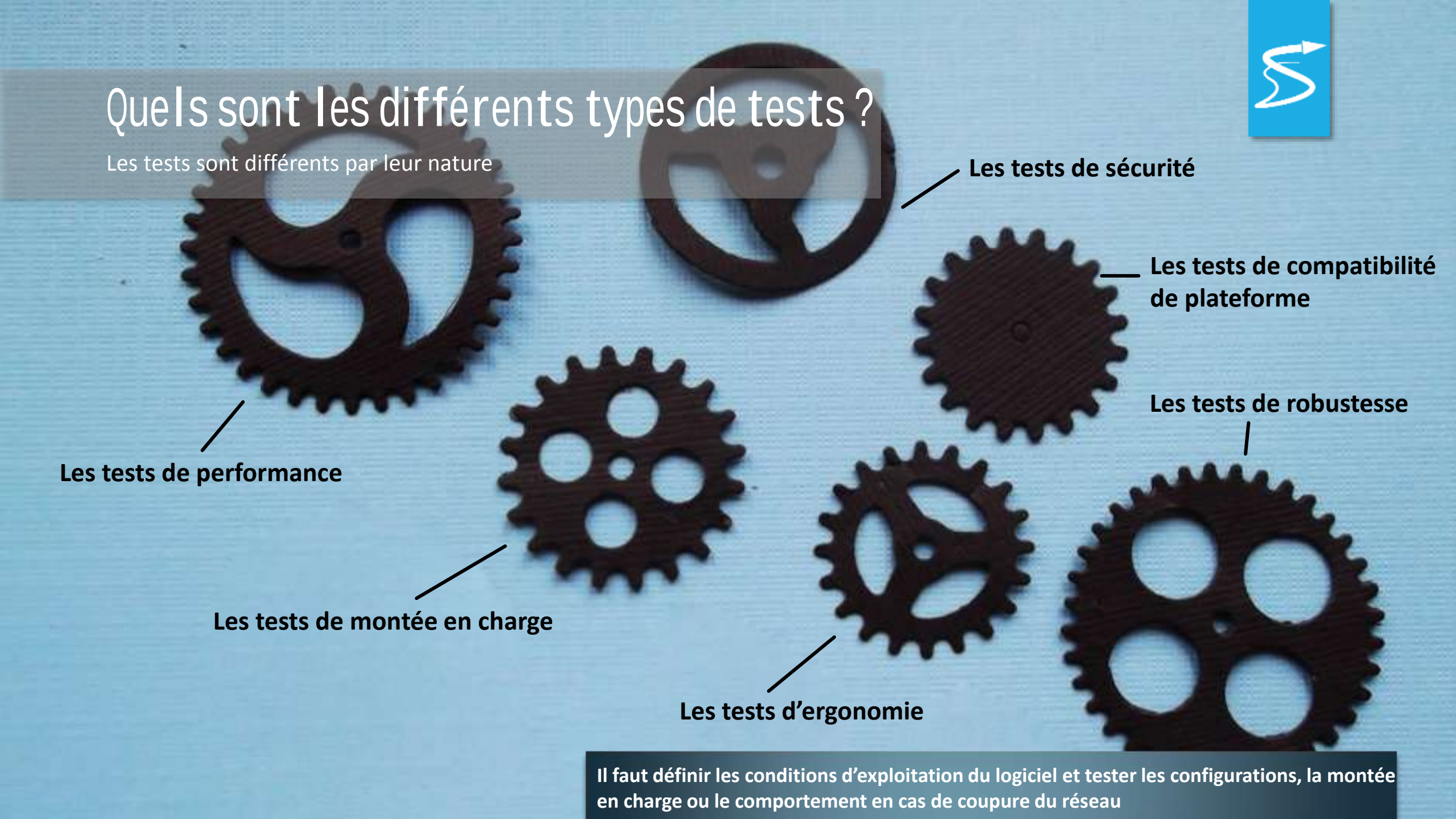
Les tests d'acceptation et fonctionnels

Les tests d'interface graphique

Ces tests peuvent facilement faire l'objet d'une automatisation

Quels sont les différents types de tests ?

Les tests sont différents par leur nature



Les tests de sécurité

Les tests de compatibilité de plateforme

Les tests de robustesse

Les tests de performance

Les tests de montée en charge

Les tests d'ergonomie

Il faut définir les conditions d'exploitation du logiciel et tester les configurations, la montée en charge ou le comportement en cas de coupure du réseau

QU'EST-ce que la testabilité ?

Une application doit toujours être conçue de manière à être testable

Savoir quoi tester permet de savoir quoi développer

Rendre le code lisible et testable unitairement

Rendre le logiciel testable et compréhensible également pour structurer l'architecture globale de l'application

Rendre l'interface graphique consciente des tests

D'où l'importance d'intégrer les tests en amont

Quels sont les scénarios ?

Chaque type de tests nécessite des scénarios spécifiques

Les scénarios sont à définir avec les experts métier

Il est difficile d'être exhaustif, il faut savoir cibler

Une première approche consiste à définir ce que l'on chercher à tester

- Composant
- Fonctionnalité
- Performance
- Tenue en charge
- Etc.

Un travail partiel bien ciblé permet de couvrir l'essentiel selon la règle 80/20

Quelles sont les données ?

Chaque type de test nécessite des jeux de données spécifiques

C’est une partie souvent négligée mais pourtant indispensable avec :

Du volume pour rejouer plusieurs fois les mêmes scénarios

ou

Des détails pour expliquer une règle de gestion ou la mettre en défaut

Comme pour les scénarios, les jeux de données sont à déterminer en commun avec les experts métiers

Quels sont les Environnements ?

Certains types de tests nécessitent des environnements dédiés

Il n’est pas toujours simple de pouvoir les mettre en place

En plus de l’infrastructure accueillant l’application il faut:

- Pouvoir déployer les outils de tests et de mesure
- Avoir des données représentatives dans les bases

Le Cloud peut être une vraie aide dans certains cas

QUELS SONT LES CRITERES D'ACCEPTATION ?

Comment bien interpréter les résultats du test ?

Comment identifier qu'un test a réussi ou échoué ?

Quels indicateurs souhaitons nous surveiller ?

Quelle est la pertinence du test effectué ?

Comment mesurer la qualité des tests ?

Nous les plaçons au cœur du processus, nous devons appréhender leur qualité

La couverture de code, une bonne mesure ?

La tendance du pourcentage de tests verts

Les indicateurs de performances pour les tests de charge

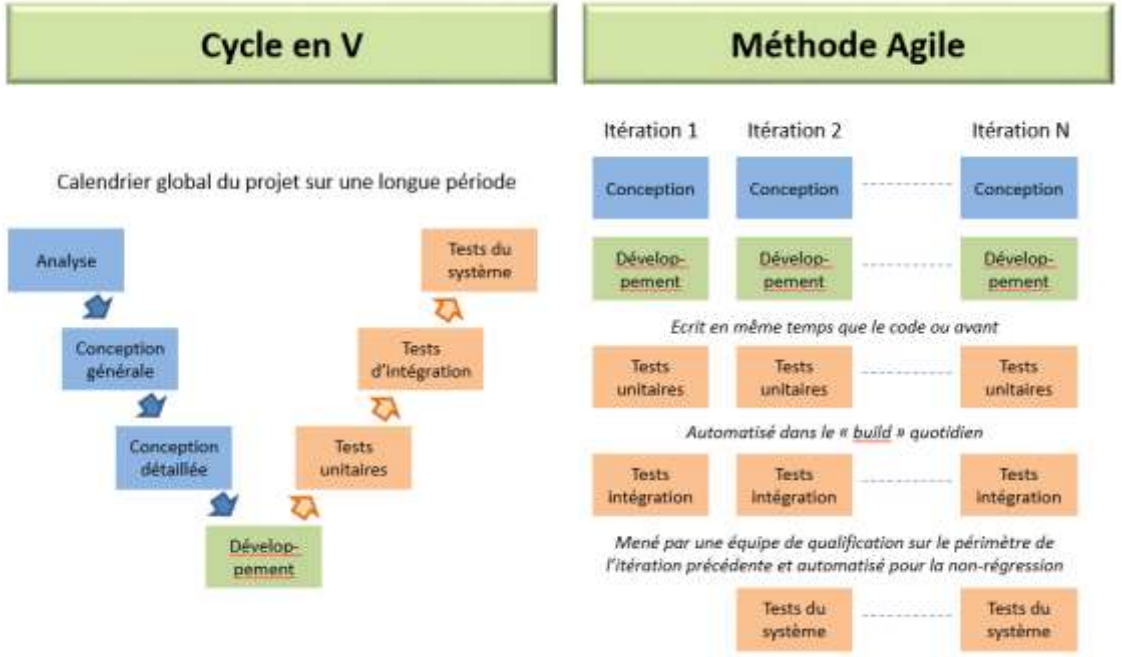
Mettre le test au centre de la problématique projet

Une évidence, quelle que soit l'approche !

Dans un cycle en V les tests arrivent à la fin

Dans une méthode agile ils doivent être présents en permanence dès le début

Dans les 2 cas nous pouvons impliquer tous membres du projet dans les tests



Les tests font partie intégrante du projet, quelle que soit la méthode, impliquons toute l'équipe !

Mettre le test au centre de la problématique projet

Penser aux tests en amont est un facteur clé de réussite !

Sensibiliser et organiser **l'équipe** autour des tests avec un rôle bien précis pour chacun des acteurs :

Les **experts métiers**

Les **développeurs**

Les **testeurs**

La collaboration entre les experts fonctionnels et les développeurs est le secret d'un logiciel réussi

Mettre le test au centre de la problématique projet

Penser aux tests en amont est un facteur clé de réussite !

Les tests comme **spécifications** :

les tests sont un bon moyen de spécifications et un support d'échange entre tous les acteurs

Les tests comme **documentation** :

si les tests sont suffisamment clairs et automatisés, ils peuvent devenir une documentation vivante

Mettre des exemples dans les spécifications, automatiser des tests, partager le plan de tests

Mettre le test au centre de la problématique projet

Faire vivre les tests durant le projet !

Les tests vivent comme le code source d'un logiciel

Ils doivent être versionnés et historisés

Nous pouvons utiliser un « source control », des branches

Utilisez un contrôle de code source type TFS

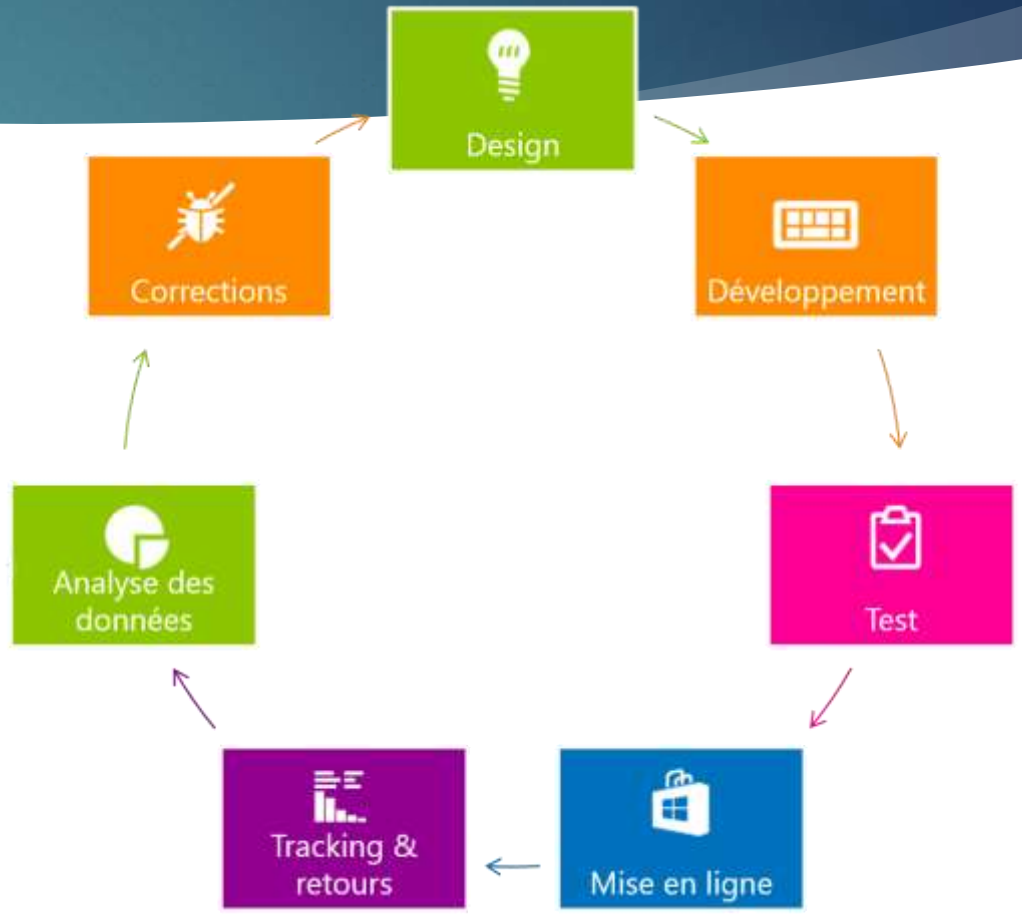
Mettre le test au centre de la problématique projet

Intégrer les tests et leur exécution dans le cycle de vie de l'application

Idéalement avoir un processus d'intégration continue qui lance les tests

Automatiser le plus de tests possibles pour que le fait de les lancer ne « coûte » rien

Pouvoir jouer une version des tests correspondant à la bonne version du code



Un test consolidé dans un script re-jouable à l'infini est un investissement

Mettre le test au centre de la problématique projet

Faciliter l'évolution des tests et l'implication de tous

Impliquer toute l'équipe

La rédaction des cas de tests doit être itératif

Les besoins pouvant évoluer il faut pouvoir changer les tests

Faciliter la maintenance des tests et l'automatisation

Les méthodes agiles préconisent une collaboration permanente et des tests en continu

En pratique chez SoftFluent

En tant qu'éditeur, experts en développement, la pratique du test logiciel est dans nos gènes !

Tests Unitaires et Test Driven Development (TDD)

Nos développeurs écrivent les tests unitaires

Nous pratiquons TDD dès qu'il a une vraie valeur ajoutée

Le refactoring fait alors partie intégrante de notre méthode

Le Test Driven Development (TDD est une technique de développement de logiciel qui préconise d'écrire les tests unitaires avant d'écrire le code source d'un logiciel. (Wikipedia)

En pratique chez SoftFluent

En tant qu'éditeur, experts en développement, la pratique du test logiciel est dans nos gènes !

Tests d'intégration et d'acceptation

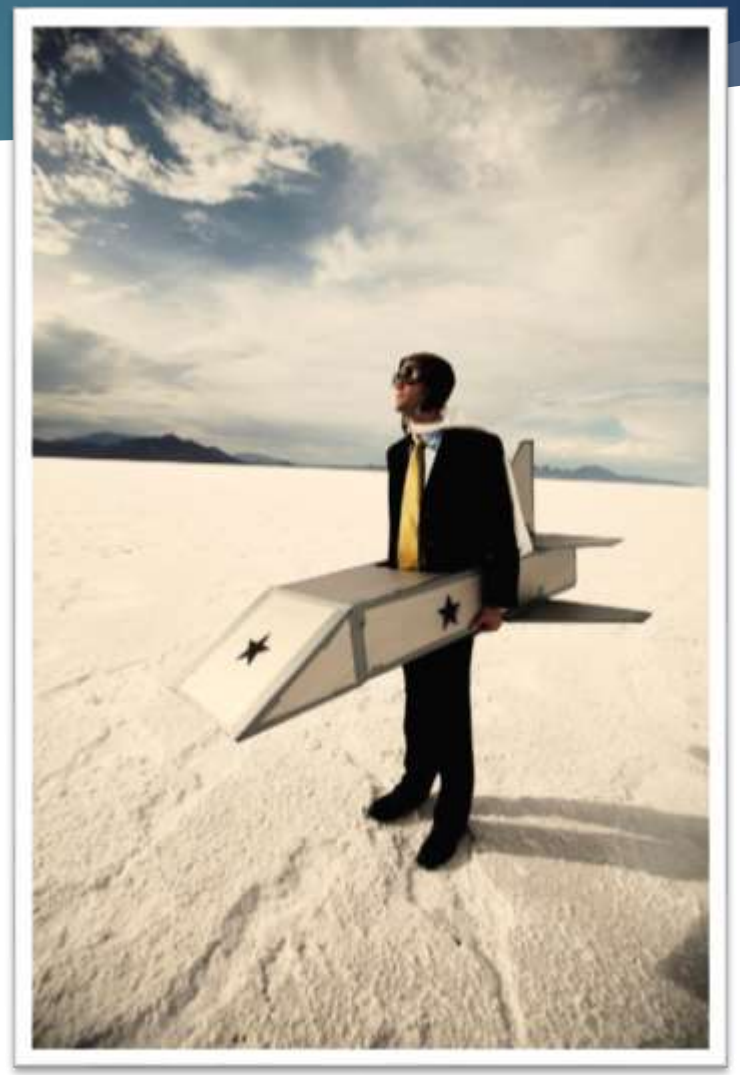
Ils nous aident à définir le besoin

Ils nous permettent de tester l'application globalement,

base de données incluse

Ils peuvent être écrits par les experts métier,

mais nous les accompagnons dans la méthodologie



En pratique chez SoftFluent

En tant qu'éditeur, experts en développement, la pratique du test logiciel est dans nos gènes !

Tests d'interface graphique

Nous les réalisons à la fin du cycle de développement

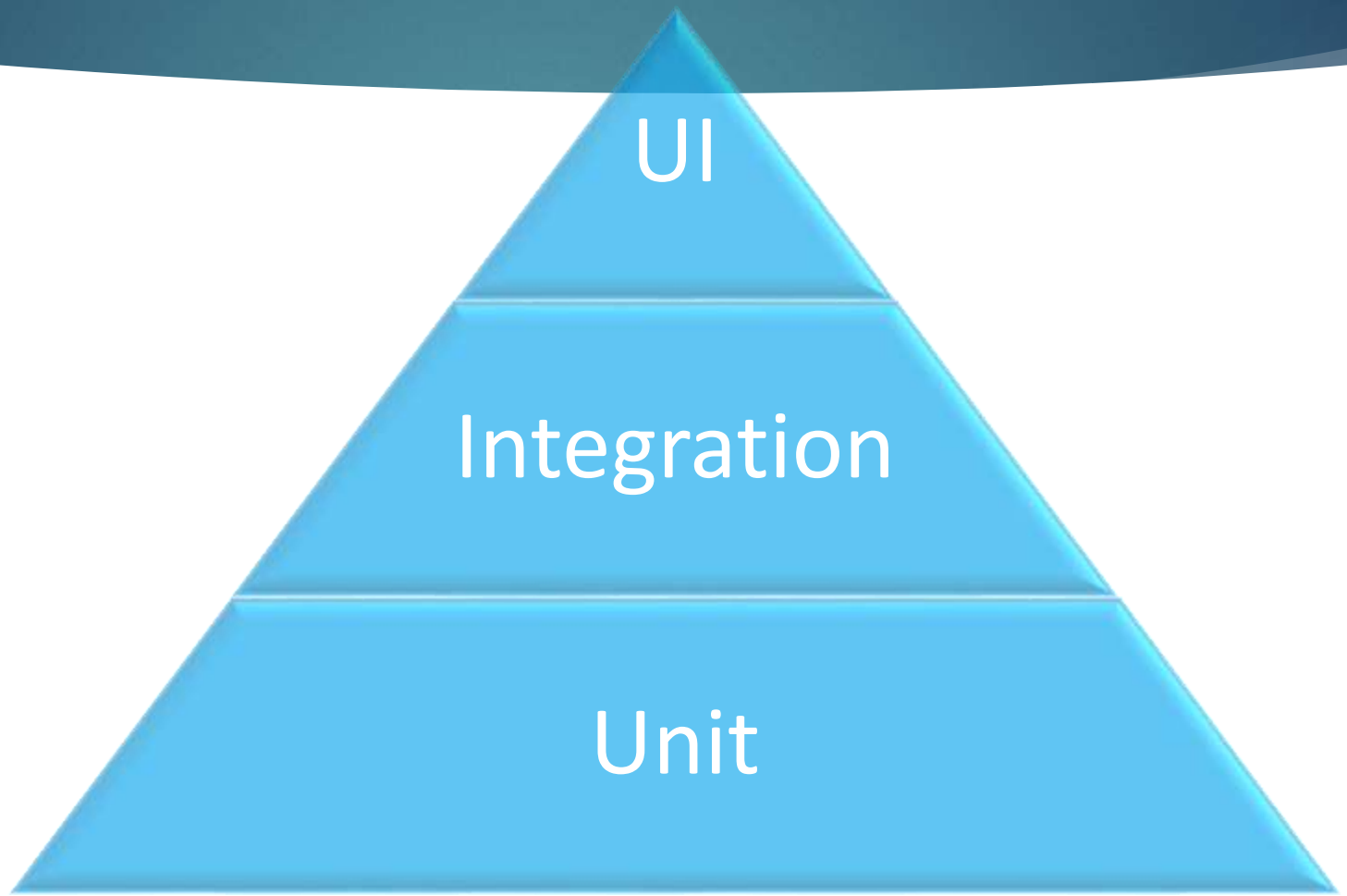
Ils nous permettent de détecter les régressions

Nous utilisons des composants existants

Nous aussi avons créé notre propre outil pour simplifier ce travail : SoftFluent Automation

En pratique chez Softluent

En résumé



En pratique chez SoftFluent

Et ce n’est pas fini…

Il reste les tests de performance qui permettent de garantir une utilisation efficace de l’application

COMMENT FAIRE LORSQUE L'ON A UN LOGICIEL EXISTANT ?

C'est au cas par cas et essentiellement sur l'interface graphique

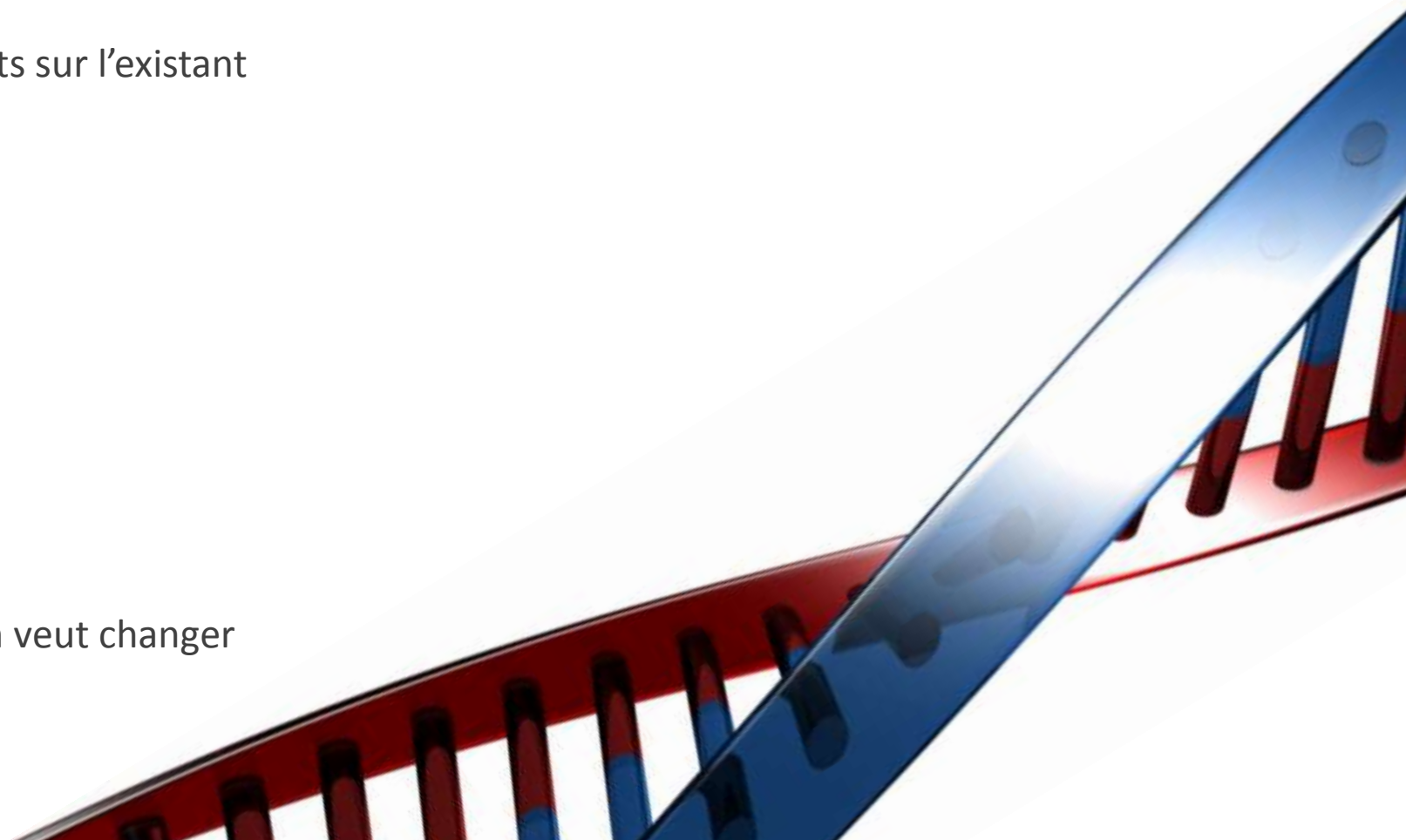
Ne rien tenter avant d'avoir écrit des tests sur l'existant

Ecrire des tests macroscopiques

- Sur l'interface
- Sur une portion de code

Refactorer ou réécrire la partie testée

Ne pas surinvestir mais cibler ce que l'on veut changer



COMMENT FAIRE LORSQUE L'ON A UN LOGICIEL EXISTANT ?

On fait du test de « couverture »

Procéder par étapes atteignables

Limiter à une fonctionnalité à la fois

Ne pas couper un flux business

Intégrer le nouveau code à l'existant

Avoir un feedback rapide

Les outils

Les outils nous aident mais ne font pas tout à notre place

Visual Studio et TFS

- **Tests unitaires avec MSTest, Nunit etc.**
- **Tests d'interface avec Coded UI Test**
- **Tests de performance**
- **Automatisation**

Les outils

Les outils nous aident mais ne font pas tout à notre place

Tests d'intégration / d'acceptation

SpecFlow

FitNesse

Cucumber

Les outils

Les outils nous aident mais ne font pas tout à notre place

Tests d'interface

- Selenium pour le web, **SoftFluent Automation**

Pour aller plus loin

Quelques références sur les tests

Specification by Example, *Gojko Adzic*

Test Driven .NET Development with FitNesse, *Gojko Adzic*

Working Effectively with Legacy Code, *Michael Feathers*

Livres et Blogs de *Martin Fowler, Robert C. Martin, Kent Beck*

SoftFluent Automation : http://www.softfluent.com/docs/softfluent/softfluent-automation.pdf?sfvrsn=2

MERCI

www.softfluent.fr

blogs.softfluent.com info@softfluent.com

SoftFluent