# 2020, Odyssée de l'Automatisation

Vittorio Capellano

*Test Practice Leader*

Journée Française des Tests Logiciels

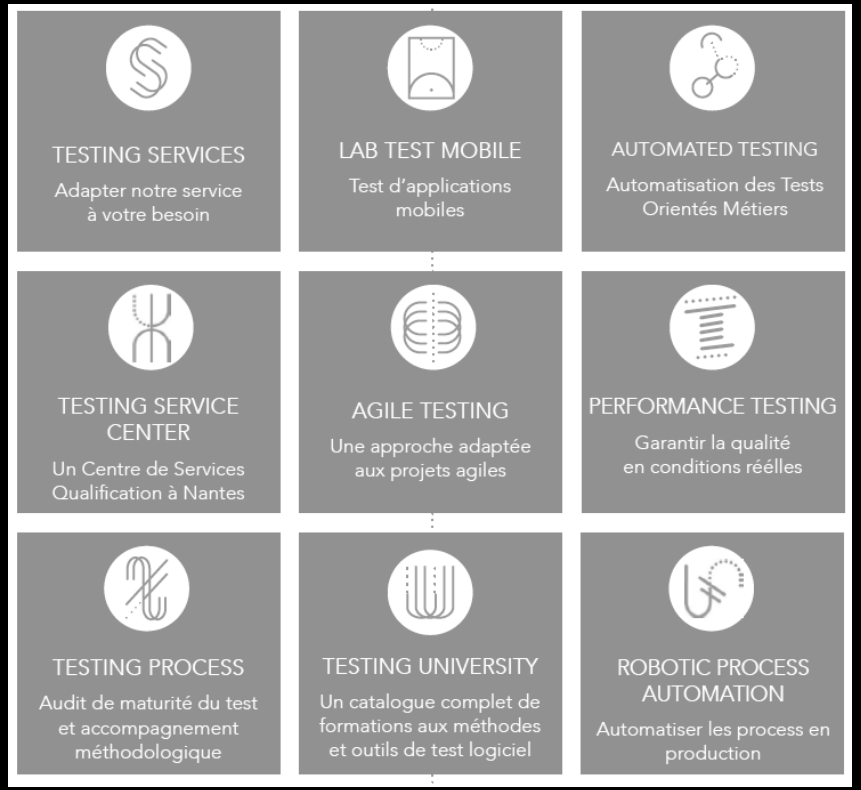


acial

# A propos d'Acial

- Pure Player du Test Logiciel
- Plus de 20 ans d'expertise au service de la qualité logicielle
- Positionnement unique de « Tiers de Confiance »
- Accompagnement de proximité
- + de 450 références multi secteurs pour tous types de projets

| 25 | 300 | 80% | 600 | 100 |
|---|---|---|---|---|
| M€ de CA | Consultants | Certifiés ISTQB | Projets de test | Recrutements prévus |




acial

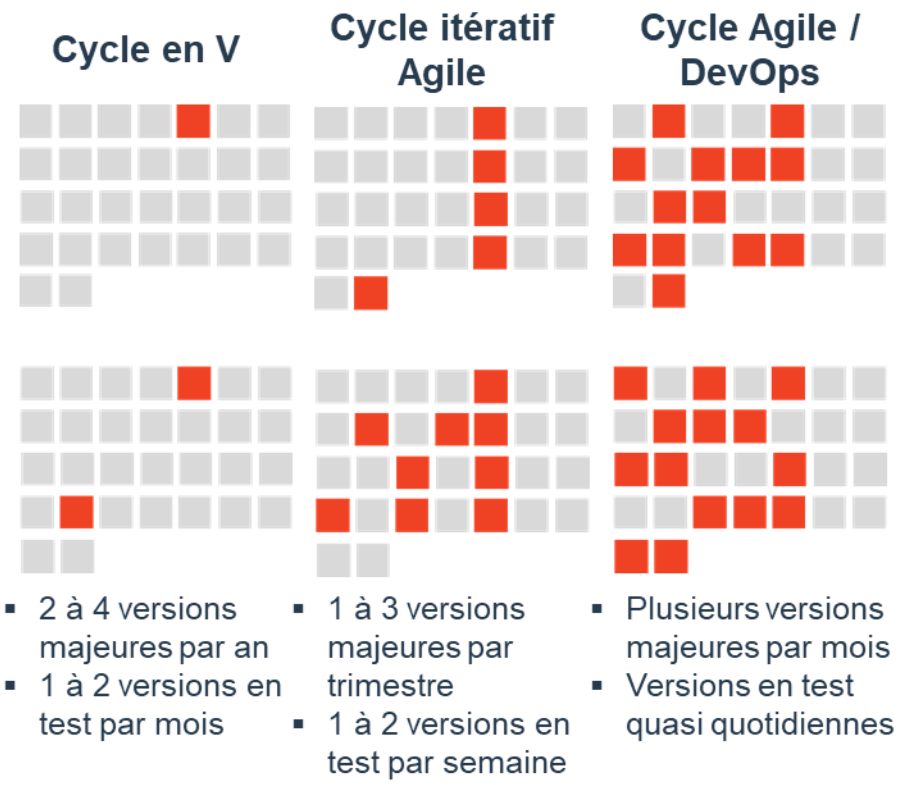
# Le voyage

1. Les fondamentaux
2. Les ressources : testeur ou développeur ?
3. Les outils : code ou codeless ?
4. Les pratiques

acial

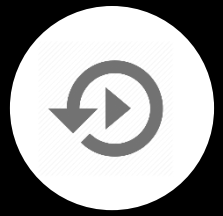
# Les fondamentaux

- **La fréquence des livraisons de versions s'est considérablement accélérée**
- **Les tests doivent donc être effectués plus souvent et plus vite**
- **Ils ne doivent cependant pas compromettre la qualité**
- **La qualité devient l'affaire de tous**

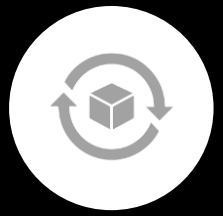


# Les fondamentaux

L'automatisation des tests est devenue une réponse logique à l'accélération de la fréquence des versions et les pratiques et outils ont accompagné cette évolution



**Capture – Rejeu**
Enregistrer (ou coder) les actions du testeur pour exécution immédiate



**Framework (data driven, keyword…)**
Variabiliser les données et actions afin d'exécuter N fois un même script

**Industrialisation & Test Continu**
Optimiser les temps d'exécution des suites de tests automatisées

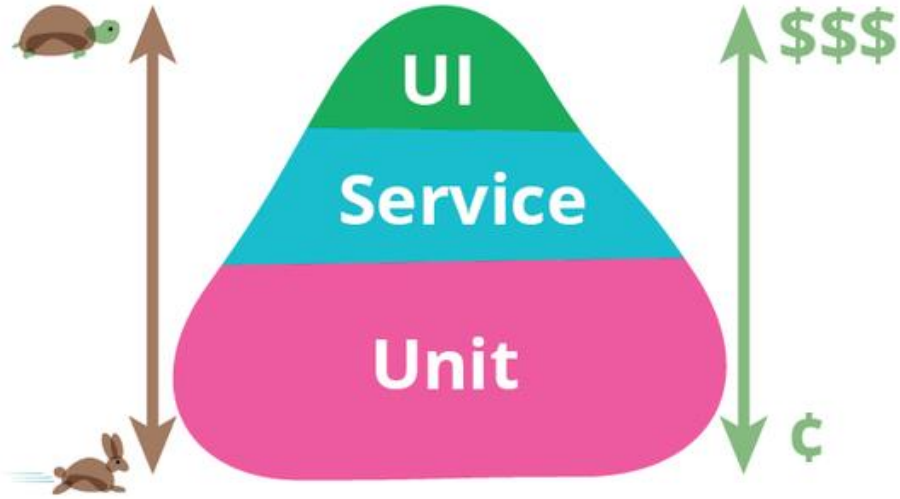
**Machine Learning / AI**
Assurer une exécution robuste face aux changements du SUT ou de l'environnement

Réutilisabilité → Performance → Maintenabilité

acial

# Les fondamentaux

La pyramide des tests est un modèle incontournable pour appréhender la problématique de l'automatisation des tests

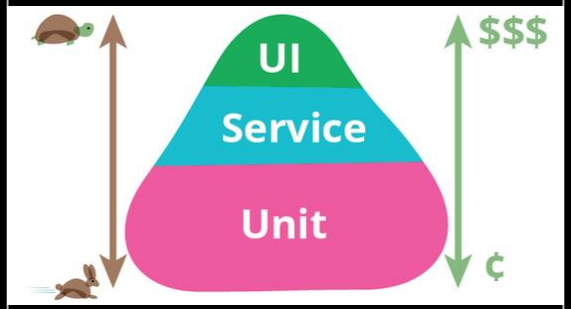



https://martinfowler.com/bliki/TestPyramid.html

# Les fondamentaux

La 1e version (Mike Cohn) illustre la répartition des tests automatisés selon les niveaux

- Tests unitaires automatisés au fur et à mesure de l'écriture du code
- Tests fonctionnels automatisés sur l'IHM pour une vision d'ensemble du fonctionnement
- Tests de logique métier au niveau des services (API) sans passer par l'IHM




La 2e version (Martin Fowler) tient compte des considérations entre coûts et rapidité d'exécution

- Plus le niveau du test est élevé, plus le temps d'exécution augmente et le test devient fragile
- Les coûts d'exécution et maintenance deviennent plus importants

*Le choix du modèle d'automatisation doit résulter d'une analyse conjointe de facteurs :*

- *Humains : ressources et compétences*
- *Techniques : technologies et outils*
- *Pratiques : approches et bonnes pratiques*

acial

# Le voyage

1. Les fondamentaux
2. Les ressources : testeur ou développeur ?
3. Les outils : code ou codeless ?
4. Les pratiques

acial

# Les ressources : testeur ou développeur ?

**Développeur**

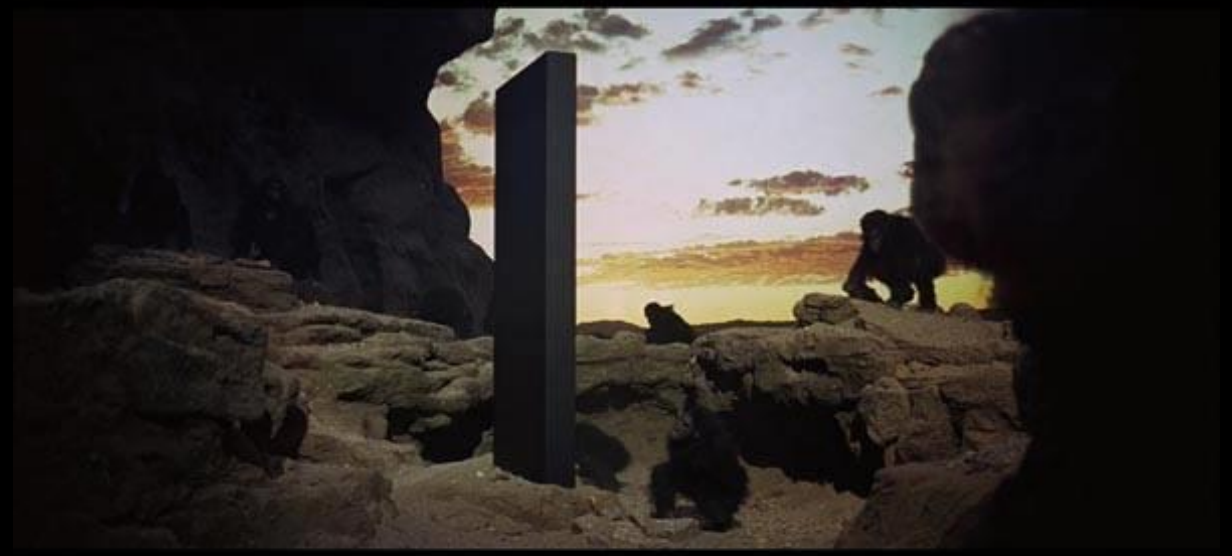
**Forces**
- Bonne connaissance de la programmation
- Affinité avec les problèmes techniques
- Apprentissage rapide

**Limites**
- Tendance à aller trop vite
- Intérêt pour le test
- Maintien de l'effort sur du long terme

**Testeur**

**Forces**
- Ressource dédiée et formée au test
- Appétence pour la recherche de bugs
- Tendance à aller dans le détail

**Limites**
- Intérêt pour la technique et la programmation
- Courbe d'apprentissage plus lente



*Le profil idéal de l'automaticien est souvent à la croisée des deux*

acial

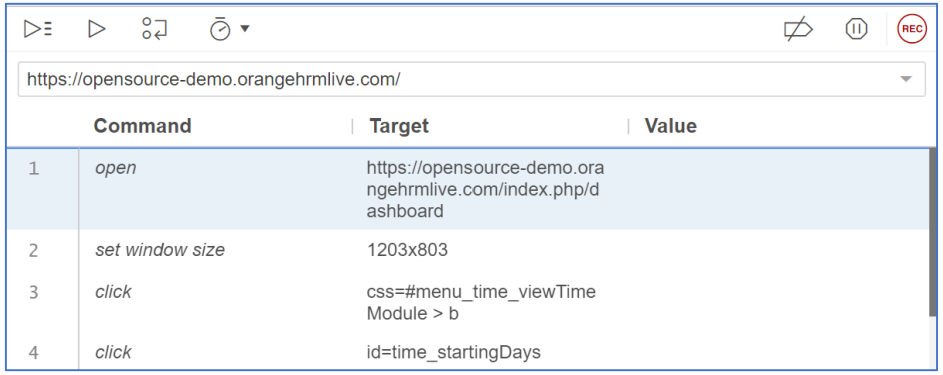
# Les compétences

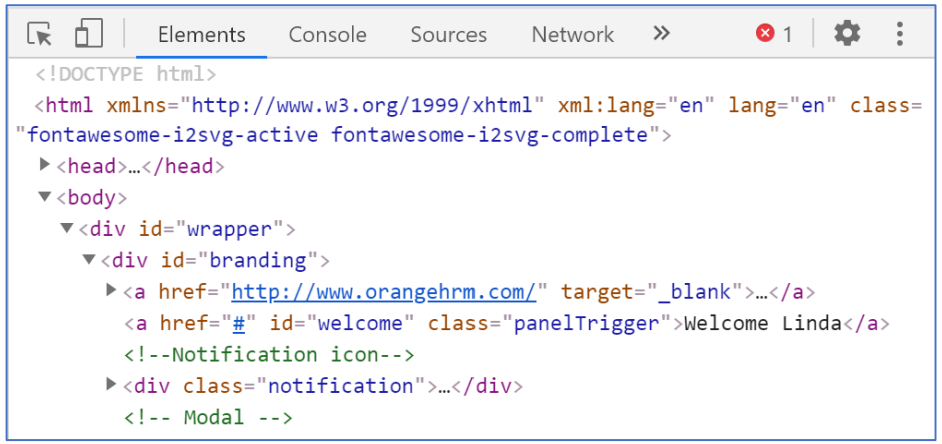
Que faut-il maîtriser vraiment quand on parle d'automatisation ?

**1. Outillage**
Savoir utiliser les fonctionnalités de base d'un outil d'automatisation

**2. Technologie**
Se repérer dans les couches techniques de l'application : objets HTML (DOM), interfaces (API)…

**3. Programmation**
Se familiariser avec un langage de développement pour appréhender les fonctionnalités avancées

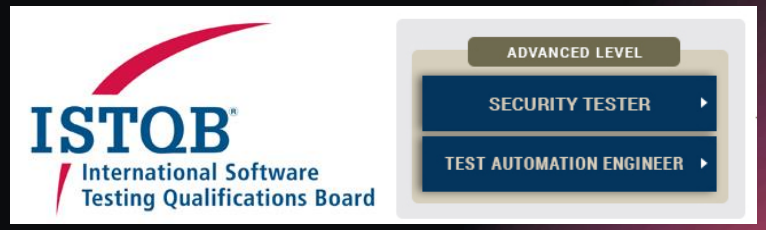



```
public static void setBrowserConfig() {
    System.out.println(browser);
    if (browser == "Firefox")
    {
        System.setProperty("webdriver.gecko.driver", "D:
        driver = new FirefoxDriver();
        System.out.println("Firefox driver initiated");
    }
```

# Les compétences

## Se former






Comment aborder sereinement la prise en main de l'automatisation ?

## S'informer












## Echanger



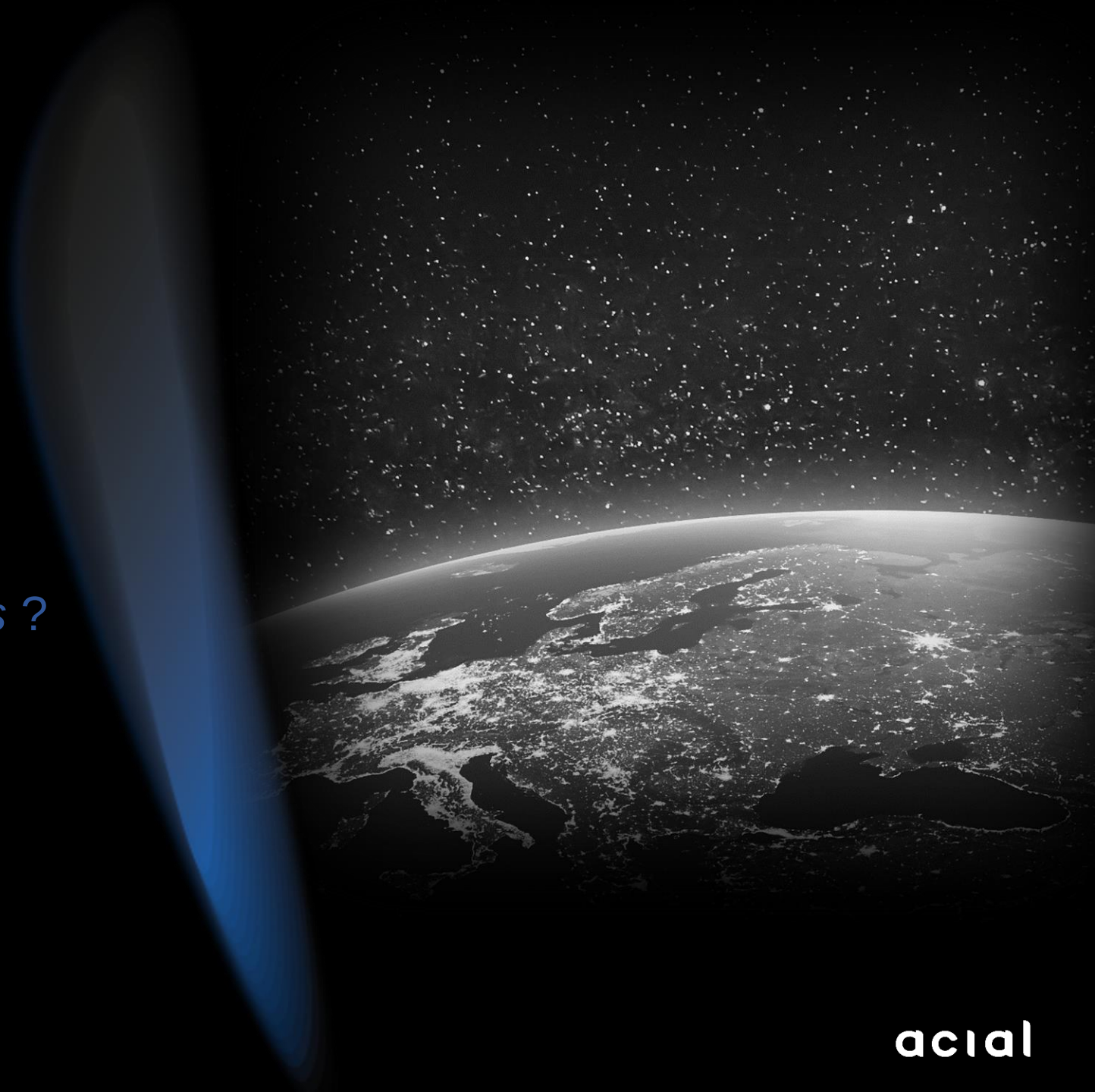




acial

# Le voyage

1. Les fondamentaux
2. Les ressources : testeur ou développeur ?
3. Les outils : code ou code*less* ?
4. Les pratiques



acial

# Choisir le bon outil : code ou code*less* ?

Choisir le(s) bon(s) outil(s) est une étape cruciale pour la réussite du projet d'automatisation

Le contexte actuel voit une multiplication d'outils et de frameworks avec une recherche d'innovation permanente dans les fonctionnalités proposées

Le choix doit résulter d'une analyse combinée de plusieurs paramètres

**Economique**
- Investissement initial et maintenabilité long terme
- Open source vs éditeur

**Technique**
- Couverture du périmètre applicatif
- Environnement technique
- Fonctionnalités avancées

**Humain**
- Ressources impliquées et niveaux de compétences
- Adéquation avec le niveau de technicité de l'outil

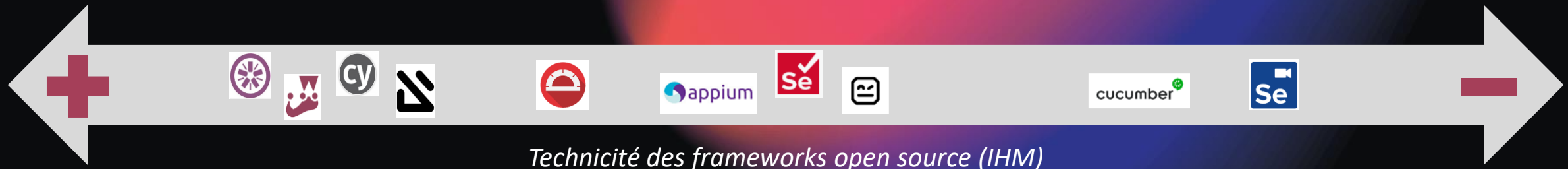
acial

# Choisir le bon outil : code ou code*less* ?

Quel que soit le niveau / type d'automatisation, les outils se rangent dans l'une de ces catégories.

Connaître les avantages et inconvénients ainsi que la technicité requise est un préalable à leur mise en œuvre.

| Code (scripting) | | Codeless |
|---|---|---|
| ▪ Adaptés à tous les niveaux de la pyramide de test<br>▪ Facilitent la mise en place de framework<br>▪ Rapidité d'exécution des scripts<br>▪ Adaptation rapide à l'usine d'intégration continue | **Avantages** | ▪ Simplicité d'utilisation<br>▪ Accessibles aux testeurs<br>▪ Couverture de l'IHM garantie<br>▪ Applicables tout au long du projet |
| ▪ Accès réservé aux développeurs<br>▪ Doivent être mis en place dès le début du projet<br>▪ Pas toujours d'interaction avec l'IHM | **Inconvénients** | ▪ Moins bien adaptés aux niveaux bas de la pyramide<br>▪ Temps d'exécution plus long<br>▪ Fragilité face aux changements d'environnement |

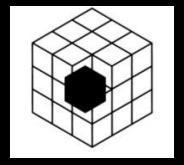



*Technicité des frameworks open source (IHM)*

acial

# Choisir le bon outil : code ou code*less* ?

Certaines fonctionnalités deviennent des critères de choix importants



**Gestion intelligente des objets**

*Référentiel, localisateurs intelligents…*

**Réutilisabilité des tests**

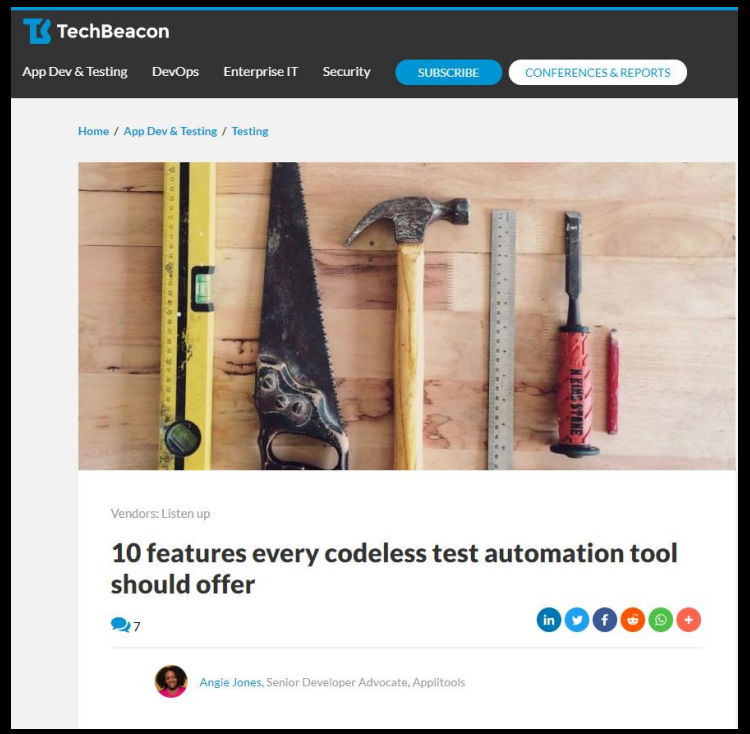
*Steps génériques, mots-clé, exécution parallèle…*


**Facilitation de la programmation**

*Boucles, conditions, attentes…*

**Test visuel**

*Capture et comparaison d'images*




acial

# Le voyage

1. Les fondamentaux
2. Les ressources : testeur ou développeur ?
3. Les outils : code ou codeless ?
4. Les pratiques

acial

# Les pratiques

## 1. Ne pas se précipiter

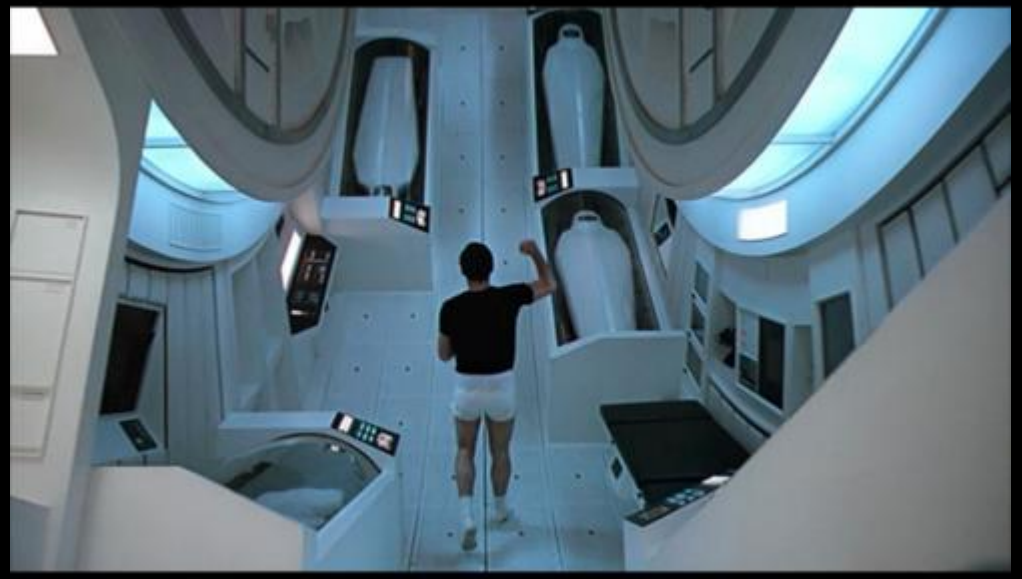
- Réfléchir à la maturité des tests manuels
- Identifier les tâches répétitives
- Organiser le projet d'automatisation

**Pilotage**

- Choix structurants sur l'outil et l'organisation
- Planification de l'activité
- Capitalisation des bonnes pratiques
- Maintien de l'effort sur le long terme
- Justification du ROI

**Réalisation**

- Mise en place des outils et des pratiques
- Création, calibration et maintenance des scripts automatisés
- Organisation de l'exécution
- Suivi et résolution des non conformités



*L'activité d'automatisation doit être envisagée sur le long terme*

acial

# Les pratiques

## 2. Choisir le bon niveau pour automatiser

Appréhender les avantages et inconvénients des différentes approches combinant outils et ressources afin de choisir son modèle de « pyramide »

| Approche | Avantages | Inconvénients/Risques |
|---|---|---|
| Automatisation de tests à partir de l'IHM | ▪ Moindre impact sur le de développement<br>▪ Couverture d'un référentiel existant de TNR<br>▪ Faible technicité requise | ▪ Efficacité liée à l'outil<br>▪ Coût de mise en place assez important (outil + ressource dédiée)<br>▪ Expertise nécessaire |
| Automatisation des test sur les API | ▪ Intéressant pour le développement<br>▪ Moins technique que le code<br>▪ Exécution plus rapidité que l'IHM | ▪ Connaissance des protocoles<br>▪ Couverture des outils<br>▪ Plus technique due l'IHM |
| Automatisation de tests unitaires sur le code | ▪ Rapidité de création des scripts<br>▪ Rapidité d'exécution<br>▪ Rapidité de détection et correction des bugs<br>▪ Faible coût (outillage de développement) | ▪ Effort de développement<br>▪ Efficacité hétérogène<br>▪ Non applicable sur un existant<br>▪ Couverture limitée au code |

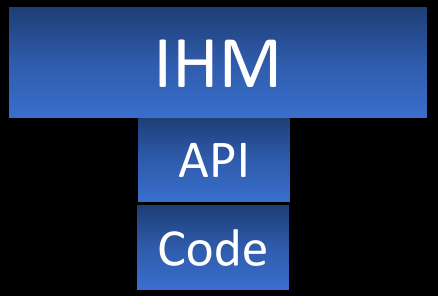
*Les approches d'automatisation sont complémentaires*

acial

# Les pratiques

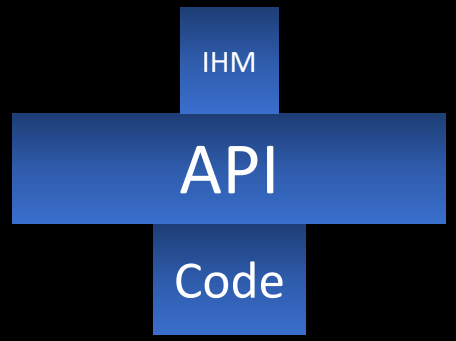
## Cas 1

- Applications Web existantes
- Peu de documentation sur les règles de gestion
- Equipes de réalisation externes



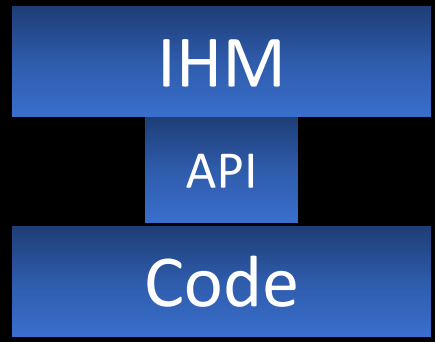

## Cas 2

- Application historique type ESB avec IHM de supervision
- Interfaces standardisées et documentées
- Equipes de réalisation externes




## Cas 3

- Refonte d'application legacy en technologie Web
- « Squad » agiles de réalisation internes
- Automatisation prise en charge par un développeur

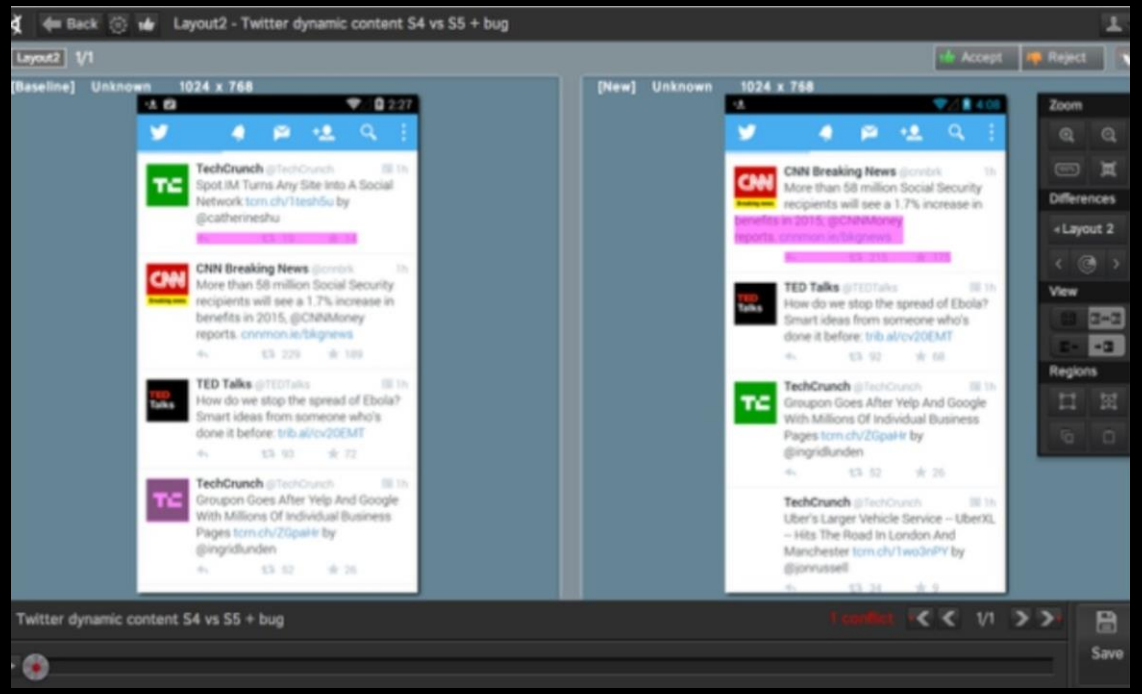



acial

# Les pratiques

## 3. Ne pas oublier l'IHM

- Certains bugs ne sont que visuels : décalages, superpositions…
- Ils ne seront pas détectables par un script même s'il s'exécute sur l'IHM
- Ils seront visibles à l'œil nu, mais pas forcément sur tous les environnements : devices, navigateurs…
- Il ne s'agit pas non plus de comparer des images au pixel près



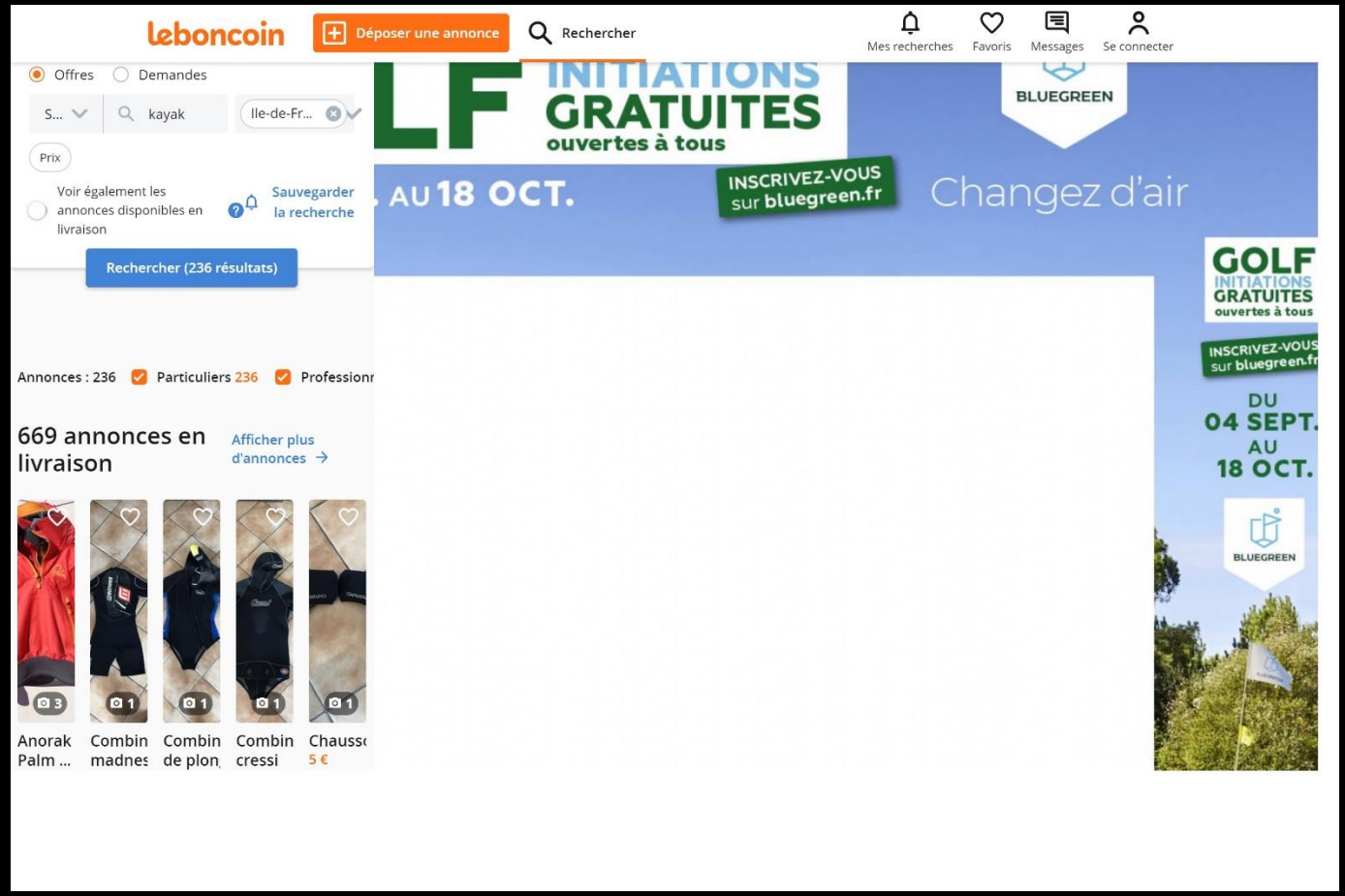

*Automating visual software testing - Applitools ©*

*Exploiter les fonctionnalités proposées par des éditeurs et frameworks open source pour combiner intelligence artificielle et reconnaissance d'images afin d'automatiser les tests visuels*

acial

# Les pratiques

Cas typique où l'automate a été capable de trouver tous les éléments de la page et le résultat du cas de test est « Pass »…



# Les pratiques

## 4. Optimiser la structure des scripts

- Les premiers scripts automatisés seront souvent complexes et rigides, rendant difficile une exécution fréquente
- Un script automatisé doit posséder certaines caractéristiques de qualité
- Comme pour un projet de développement, le refactoring des tests automatisés est une bonne pratique

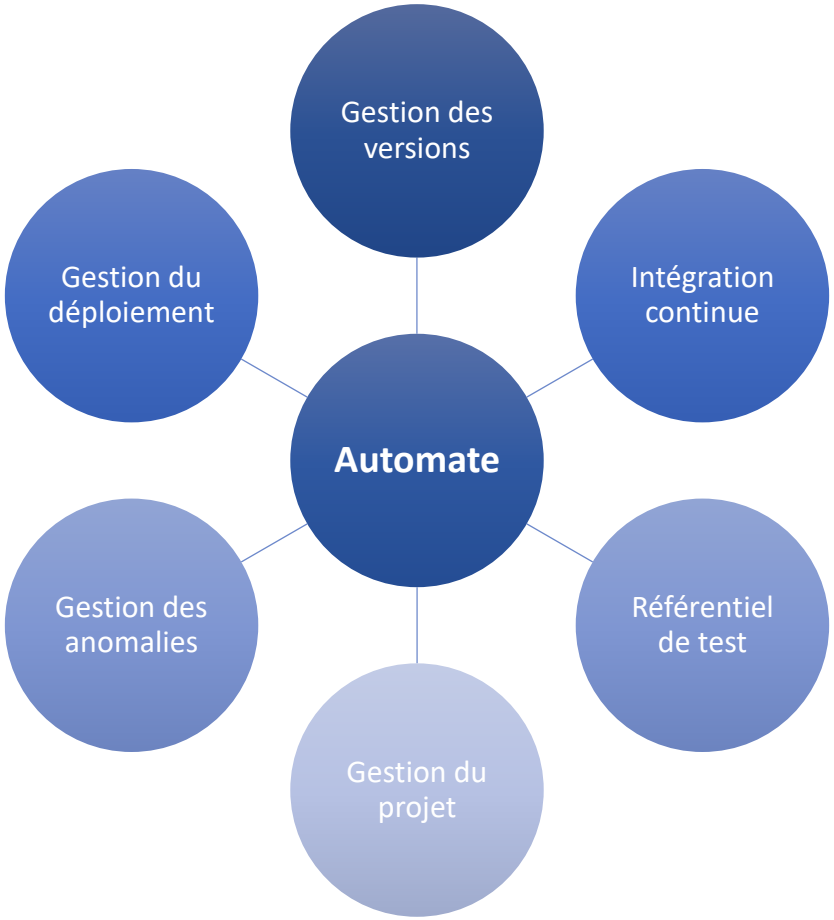
*Un test automatisé non optimisé sera toujours préférable à aucun test automatisé…*


*Richard Bradshaw / Mark Winteringham, MoT, SeConf 2019*

# Les pratiques

## 5. Anticiper l'intégration de l'outil dans la chaîne de fabrication

- Un script automatisé ne peut pas être conçu pour s'exécuter de façon isolée
- Il doit pouvoir être exécuté à l'identique sur de multiples environnements
- Dès la conception, et parfois dès le choix de l'outil, il faut se poser la question des environnements à couvrir
- Choisir un outil qui communique bien avec l'écosystème des outils de l'organisation



# Les pratiques

## 6. Adopter une approche collaborative (BDD, ATDD…)

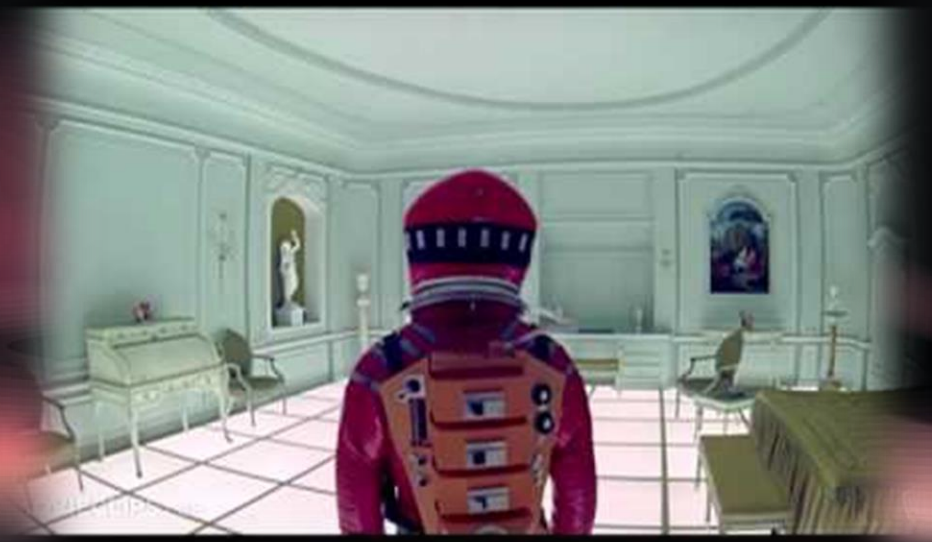
- Eviter de travailler en silo
- Favoriser la collaboration autour de l'effort d'automatisation
- S'appuyer sur des outils moins techniques
- Renforcer le lien entre testeur et développeur

*Les approches collaboratives comme le BDD créent des conditions favorables à un cercle vertueux qui engage toute une organisation dans son effort d'automatisation*

acial

# Arrivés à destination…

- Les réponses à la problématique de l'automatisation des tests sont nombreuses
- La réflexion autour du tryptique « Man – Machine – Method » permet d'aborder le problème sereinement
- L'important est de s'inscrire dans un effort continu afin de bénéficier des avantages
- S'instruire, expérimenter et innover !!!



acial

# Merci pour votre attention