

Automatisation des Tests

L'automatisation des tests fonctionnels est une étape clé dans les processus d'industrialisation de la conception et de la validation de logiciels. Apportant de réels avantages, en terme de productivité et de fiabilité, l'automatisation des tests nécessite toutefois une approche spécifique et pragmatique. Cette approche permet de prévenir les nombreux risques inhérents à l'automatisation et d'atteindre les objectifs de test (non-régression) tout en assurant un retour sur investissement (ROI).

Les principaux bénéfices de l'automatisation

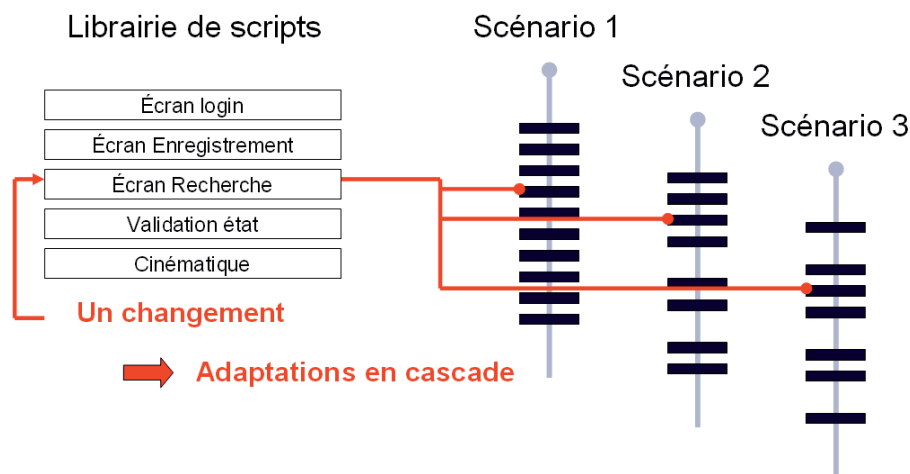
// Consistance

L'automatisation assure une véritable consistance aux tests. En effet, un cas de test se déroule toujours selon le même scénario, consomme des données spécifiques connues d'avance, valide des résultats attendus. Par ce fait, la fiabilité d'un test est garantie et, surtout, les éventuels problèmes peuvent être reproduits.

// Productivité

L'utilisation de composants de scripts, sortes de briques de test assemblables, permet une réutilisation optimale des tests automatisés. Le but est de minimiser les développements d'automates tout en maximisant leur utilisation. Ce modèle de développement est défini par le concept d'automatisation Clio.

En liant le script d'automatisation avec une structure de données à consommer ou à valider et non pas aux données elles-mêmes, les cas de tests peuvent être facilement et rapidement étendus. La couverture de test s'en trouve nettement améliorée. Cette fonctionnalité est entièrement prise en charge par le **Framework d'automatisation Clio**.



Principe de la modularité

L'automatisation replace les différents intervenants (testeurs, concepteurs de tests, tests managers) au coeur du système de test. Les ressources ne sont pas mobilisées pour des tâches répétitives mais sont assignées aux activités centrales telles que l'analyse des résultats, le développement de nouveaux tests, la planification et le management.

L'automatisation seule ne peut, en tant que telle, être réellement productive que si les résultats des exécutions sont aisément accessibles, résumés, compilés et archivés. **Ces tâches doivent être complètement incluses et transparentes dans le système de test.** On optera pour des outils d'automatisation pouvant être intégrés dans une plateforme de gestion de tests.

Même si cet avantage n'est pas le plus primordial, il n'en demeure pas moins que la rapidité du déroulement des tests automatisés est nettement plus grande par rapport aux exécutions manuelles. En outre, les plateformes d'exécution peuvent être multipliées, les exécutions lancées en mode batch hors des heures ouvrées.

Les principaux risques de l'automatisation

Tout script d'automate est en fait un programme informatique qui implique une culture et une réflexion technique et algorithmique. Une certaine forme de complexité doit donc être assumée par le concepteur d'automates. La structure de l'interface graphique à tester, une mauvaise sélection des scénarios de test à automatiser (éligibilité des cas de tests), l'utilisation intensive de données dynamiques, peuvent accroître considérablement cette complexité. Ces problèmes peuvent être nettement minimisés en adoptant une méthode et une analyse avant tout travail de développement – **une démarche d'automatisation formalisée** – et, ensuite, en définissant un cadre de développement clair et structuré. Toute complexité résiduelle sera, au maximum, cachée au concepteur d'automates par l'utilisation de fonctions de bibliothèques et autres outils – **le Framework d'automatisation**.

// Efforts de maintenance

Les tests fonctionnels génèrent essentiellement leurs plus-values à moyen et à long terme. En effet, le but premier des automates est de vérifier le bon fonctionnement d'une opération donnée suite aux évolutions apportées à l'application cible ou à tout autre élément susceptible d'avoir un impact sur le métier même de cette application. Il est donc **primordial** que les automates puissent accomplir leurs tests tout en résistant au maximum à ces évolutions.

Seule une architecture de conception fiable et basée sur la **réutilisation des automates** peut être à même de minimiser les impacts de ces changements. **Le modèle de développement Clio répond de manière efficace, au travers de son framework d'automatisation, aux exigences de modularité.**

// Maturité des processus

Pour rentabiliser efficacement les efforts d'automatisation, une certaine maturité dans les processus globaux de production de logiciel doit être atteinte. La planification rigoureuse du release management ainsi que la communication entre les différents intervenants sont les points capitaux de la réussite de la démarche d'automatisation.

