ÉCOLE DE TECHNOLOGIE SUPÉRIEURE UNIVERSITÉ DU QUÉBEC

MGL804 Réalisation et maintenance de logiciels

Été 2018

Projet de session

Sujet :

Top 3 des plus populaires outils pour chaque Catégorie d'outils de tests offerts au Mainteneur aujourd'hui

Par

abdellaziz SALAHADDINE – SALA20059003

Dirigé par : Professeur Alain April

MONTRÉAL, LE 30 JUIN 2018

Résumé

Ce rapport technique est réalisé dans le cadre d'un travail de fin de session pour le cours MGL804 s'intitulant Réalisation et maintenance de logiciel a l'École de technologie supérieure (ETS). Le sujet abordé dans ce document était parmi les propositions présentent dans le livre " Améliorer La maintenance du logiciel (Abran & April 2016) et porte sur "Top 3 des plus populaires outils pour chaque Catégorie d'outils de tests offerts au Mainteneur aujourd'hui ".

Le marché présent fournis une multitude de choix d'outils de test, chacun avec des caractéristiques différentes, certains sont plus mature que d'autres qui essaient de répondre aux besoins des testeurs afin de suivre l'évolution de la technologie et ainsi pouvoir se créer une place parmi les autres fournisseurs. C'est cette concurrence qui représente le noyau de notre sujet, notre but est de définir une List d'outils de test existant sur le marche pour ensuite déduire le top 3 de ces outils selon leurs popularités chez les mainteneurs.

Ce rapport repose sur les connaissances théoriques liées au tests. Les données utilisées découlent des études précédemment réalisé par des organisations reconnus et dont les critères cohérent avec ceux qu'on a choisi. Le travail tient compte de l'aspect temporelle de sa publication dans le but de fournir des informations d'actualité dans laquelle l'étude a été réalisé.

Liste des tableaux :

Liste des figures :

Table des matières

Introduction :

Ce projet de session s'intéresse aux logiciels utilisés par les mainteneurs durant la phase de test après une modification au niveau d'un système existant. Cette étude nous offre une possibilité de définir les 3 outils les plus populaire au sein de cette communauté.

Pour ce faire, une étude littéraire sera menée sur différent outils ainsi que leurs points forts et points faibles, afin de répondre aux questions problématiques suivante : qu'elle le bon critère pour faire la différence entre les multiples outils de testes existant ? Et qu'elles sont les 3 outils de tests les plus populaires selon ce critère ?

Contrairement aux autres études dont le résultat est susceptible d'être soumis à des vérifications pour décider de son exactitude, celle ci est une comparaison qui n'est valable que pour un période dans laquelle le résultat de classement de 3 outils les plus populaire est inchangé, dans le cas contraire ce qui restera de l'étude c'est la méthodologie suivis afin de pouvoir définir les successeurs.

Une méthode sera utilisée pour permettre au mainteneur de définir l'outil qui répond le plus efficacement à leurs besoins, et qui leur permettra un meilleur résultat en tenant compte de la relation coût, effort. Le travail propose aussi une méthode semi-sélective qui permet de déterminer une liste d'outils pour arriver au résultat recherché des trois outils les plus populaires.

Description du projet :

Avec la complexité des systèmes et les difficultés auxquelles les mainteneurs font face pour réaliser leurs taches. L'idée derrière ce travail est de mettre en place une base de référence sur les outils de tests présents sur le marché, cela facilitera leurs prises de décision lors du choix de l'outil de test le plus appropries à leurs besoins.
Un inventaire des outils sera parcouru dans le document et selon les critères définis, une sélection désignera les plus populaires d'entre eux. Un essai et une démonstration permettront de décrire aussi les avantages et les inconvénients de chacun de ces outils fera l'objet de ce travail.
La portée de ce travail est conditionnelle et n'est en aucun cas une exactitude à date indéterminé, vue que certains outils peuvent évoluer par rapport à d'autre ainsi que leurs popularités, ce qui limite sa validité sur l'axe temporelle. Mais néanmoins celui-ci représente une base de connaissance sur les critères de choix de logiciels et leurs situations à un moment donné (celui de la rédaction du document).

Revue de la Documentation :

Comme dans chaque pratique chaque année des comparaisons d'outils de tests est réalisés pour orienter les testeurs aux outils qui répondent le mieux à leurs besoins. Dans mon travail je me base sur ces études déjà faites et parmi eux celle faite par Mr Anas

Ouardirhi pour pouvoir mettre à jours les outils qui ont gagner de la popularité et ceux qui sont restés Fidel a leurs réputations de leader.

Méthodologie de travail :

Je suivrai dans mon approche un enchaînement en deux étapes :

1. Identifier le critère de classification des outils de tests
    a. Trouver le critère le mieux placer pour regrouper les outils
    b. Qui sont les outils les plus présents dans la marche
    c. Définir des critères de classement
2. Définir les mesures de classement des outils
    a. Se baser sur les fourmes, et les résultats des recherches sur les moteurs de recherches
    b. Les livres et les articles spécialisées dans le même sujet de cette recherche

Processus de conception :

Les tests :

Définition :

IEEE : (A) un ensemble d'un ou plusieurs cas de test.

Cas de test : un ensemble de valeurs d'entrée, de conditions d'exécution, de résultats attendus et de conditions d'exécution, développé pour un objectif ou une condition de test particulier, par exemple pour exécuter un chemin de programme particulier ou pour vérifier la conformité à une exigence spécifique [1]

Swebok :

Le test logiciel consiste en la vérification dynamique qu'un programme fournit des comportements attendus sur un ensemble fini de cas de test, choisis de manière appropriée dans le domaine d'exécution généralement infini. [2]

En prévenant en compte la définition du IEEE 2008 on peut rajouter les procédures de test a la définition et qui contient les instructions pour l'exécution et l'évaluation des résultats des cas de test et qui dans les définitions représentent la base des tests a réalisé

Niveau de tests :

Brièvement ce sont les différentes étapes de cycle de vie du logiciel ou les tests sont effectués ils sont au nombre de 4 :

Tests unitaires :

Le niveau où chaque unité du logiciel est testée individuellement, l'objectif est de s'assurer que chaque unité du logiciel fonctionne comme prévu dans la conception. [3]

Tests d'intégration :

Le niveau ou toutes les unités individuelles sont combine et teste en tant que groupe, l'objectif de ce niveau de test est de détecter les défauts surgissant lors d'interaction entre les unités intègres. [3]

Tests système :

Ce niveau est entamé juste après la fin des tests d'intégration permettant de tester l'ensemble de system dans un environnement tel que celui chez le client avec des données réelles, permettant ainsi de vérifier si le logiciel repend aux exigences fonctionnelles et non fonctionnelles exprime par le client. [3]

Test d'acceptation :

Le but de ce niveau est d'être sûr que le logiciel est prêt à être livré et qu'il rencontre les exigences d'affaire du client, ceci permet de recevoir une acceptation définitive de la part du client. [3]

Types de tests :

Les tests sont présents sous différent type, d'ailleurs une liste exhaustive peut être mise en place pour pouvoir tous les déterminer, mais ils ne sont pas tous appliqués. Durant le cycle de vie du logiciel, des critères comme l'utilité d'un tel type de test ou d'un autre est identifier dès les premières étapes selon l'objectif à vouloir atteindre derrière, ainsi que la complexité du logiciel a réalisé qui peut déterminer son intégrité et d'aider à décider de la priorité de type de test par rapport à d'autre.

Dans ce travail on essaye de mettre le point sur les plus importants types de tests :

- Tests fonctionnels
- Test non fonctionnel
- Test d'utilisabilité
- Test d'interface utilisateur
- Test de sécurité
- Test d'intégrité de base de donne
- Test de tolérance aux erreurs
- Test de configuration et de déploiement
- Test de régression
- Test de performance

Le tableau suivant montre la validité de quelques types de test selon les quatre niveaux de test, on peut constater que certain type de test comme les tests fonctionnels, tests de performance et documentation sont valable dans les différents niveaux, alors que ceux de configuration, interface user utilisabilité et

installation ne sont valable que durant les niveaux de tests system et d'acceptation. D'autres types de tests sont inclus dans le tableau. [4]

	Test unitaire	Test d'intégration	Test system	Test d'acceptation
Test fonctionnel	Oui	Oui	Oui	oui
Test de configuration	Non	Non	Oui	oui
Test interface utilisateur	Non	Non	Oui	oui
Test utilisabilité	Non	Non	Oui	oui
Test d'installation	Oui	Oui	Oui	oui
Documentation des tests	Oui	Oui	Oui	oui
Test de performance	Oui	Oui	Oui	oui
Test de sécurité	Non	Oui	Oui	oui
Test d'évolutivité	Non	oui	oui	Non

Tableau 1: Resume des types de test logiciel

Identification de critère de classification des outils de tests

Pour le choix de critère majeur de classification des outils de test, l'étude menée dans ce travail classera les outils de tests selon leurs catégories, représentant ainsi une vue différente de celle présentée par le travail réalisé précédemment et permettant aussi de tirer une conclusion sur la vitesse d'évolution et de maturité des outils dans le marché.
Les catégories dans ce document sont :

- Outils de test fonctionnel et de régression
- Outil de test de performance
- Outils de test de mobile
- Outils de gestion de test
- Outils de gestion de défaut

La popularité d'un outil est définie suite aux avantages qu'il fournit au mainteneur, par exemple ce dernier sera plus intéressé par un outil qui lui permet de réaliser des tests pour différentes plateformes si son environnement est constitué autant que tel, mais ne verra pas la nécessité s'il utilise une seule plate-forme qu'il est sûr de ne pas changer à long terme, dans ce cas-là le mainteneur préfèrera un outil qui lui offrira plus de performance ou de facilite d'utilisation. Ceci dit les fonctionnalités joue un rôle important comme critère pour définir le positionnement d'un outil parmi d'autres.
Ceci dit les classements faites durant les études précédemment menées, les discussions dans les forums représentent un critère qui peut belle et bien être exploiter même si c'est difficile de conclure des suggestions, vue que ce travail ne bénéficie pas de la durée suffisante pour mener une recherche approfondie, mais qui sera pris en considération dans notre étude.

Les outils disponibles de tests

La liste présente si dessous visent les outils qui ont été classé les plus populaire durant la dernière année, à savoir qu'elle reste pas complète vue le grand nombre qui existe sur le marché, que ce soit des outils commerciaux ou des outils open source, je n'ai pas pris ce critère en considération, puisqu'une étude pourrais la relève pour pouvoir déterminer les outils payant sont mieux adapte au besoin que ceux qui sont open source, et est-ce que cela affecte leurs popularité au sein de la communauté.

Mais Avant de passer vers les outils de test les plus populaire, il était primordial de récolter les informations sur leurs classifications de la part des utilisateurs. Une étude menée par G2 Crowd Grid a permis d'avoir un retour sur leurs positionnements.

En regroupant les outils sous 4 catégories : les leaders, les prétendants, les plus performants et les Niches on conclut par les résultats que la plupart des fournisseurs essaient de positionner leurs produits comme des leaders, une catégorie qui a obtenu le plus grand score de vote utilisateurs vue leurs présences dans le marché, suivis par les plus performants et ce sont ceux qui manquent de maturité et essaient de développer et intégrer plus de fonctionnalités pour pouvoir rejoindre la catégorie des leaders.

Figure 1:Placement des outils par catégories

La figure suivante faite par Mrs Joe Colantonio pour le compte de Tricentis utilise 4 autre catégories : visionnaire, leaders, Challengers et niche players pour classifier les fournisseurs de 9 outils de tests, cette expérience a porté sur des sondages auprès des utilisateurs des outils, réponses de fournisseurs et des évaluations de produits. Les résultats de cette expérience étaient comme montre dans la figure.

Figure 2: classification des fournisseurs

Outils de test fonctionnel et de régression

- Selenium (outil open source)
- Katalon Studio (outil open source)
- HP unifiai Functional testing (outil commercial)
- Rational functional tester (outil commercial)
- SoapUI (outil open source)
- SilkTest (outil commercial)
- Watir (outil open source)
- TestComplete (outil commercial)
- Cucumber (outil open source)

Outils de test de performance

- LoadRunner (outil commercial)
- JMeter (outil open source)
- RPT (outil commercial)
- Silk Performance (outil commercial)
- WebLoad (outil commercial)

Outils de test mobile

- Appium (outil open source)
- TestComplete (outil commercial)

- Calabash (outil commercial)

Outils de gestion de test

- qTest (outil commercial)
- PractiTest (outil commercial)
- Zephyr (outil commercial)
- Test Collab (outil commercial)
- QAComplete (outil commercial)

Outils de test pour la gestion de défaut

- BugZilla (outil open source)
- IBM Rational ClarQuest (outil commercial)
- Mantis (outil open source)
- BugHost (outil commercial)

Les outils les plus populaires :

selenium :

Selenium est peut-être le plus populaire de ceux que je vais citer dans ma liste, c'est un Framework d'automatisation qui se basent sur plusieurs outils et plugins pour les tests d'application web. Il est connu pour sa capacité à supporter les tests performance de ces applications. Sa large communauté de développeurs active et sa qualité d'open source lui ont permis de gagner une grande popularité en plus des avantages qu'ils représentent.

Avantages :

- Open source en licence et en maintenance
- Sa communauté large et active lui permet de suivre l'évolution de la technologie
- Facilite d'intégration avec les autres outils

Inconvénients :

- Besoin d'un niveau de programmation et d'expérience pour pouvoir l'intégrer avec les autres solutions et Framework

- Un support long pour la communauté
- Besoin d'un effort pour l'intégration et la configuration

Simulation :

Vue que ce n'est pas le but du projet, les figures suivantes représentent une simple simulation de selenium, cette partie vise à intégrer selenium au sein d'un browser afin de montrer comment l'outil permet aux testeurs de créer leurs cas d'utilisation.
Pour ce cas d'utilisation, on utilise le plugin de selenium avec le navigateur FireFox, sur un scenario dans lequel on essaie de créer un cas de tests pour un site web, on a choisi amazon pour l'exemple. La première figure de cette simulation affiche l'interface de selenium, celle qu'on va utiliser en premier lieu pour appliquer la méthode d'enregistrement du cas de test et qui permet au testeur d'enregistrer les interactions de l'utilisateur avec le site web.
Durant l'enregistrement selenium insert automatiquement des commandes en se basant sur les actions que l'utilisateur entame, que ce soit un click, une valeur entrée, ou un choix d'option depuis une liste déroulante, ou un choix dans un check box ou radio bouton.

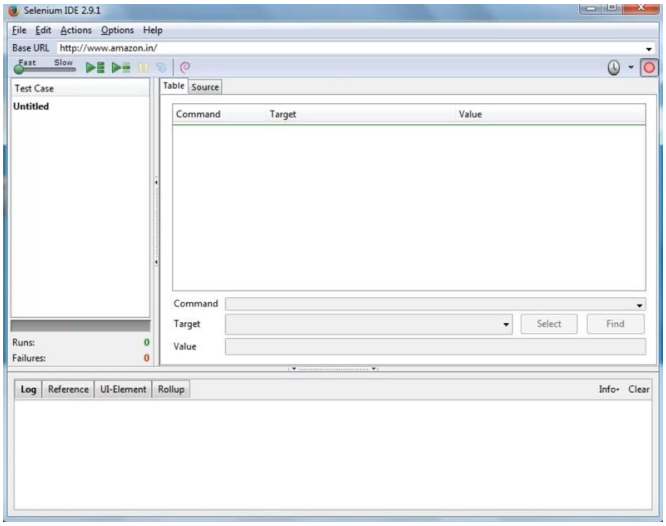

Figure 3: interface d'initiation au cas d'utilisation selenium

Après avoir enchainer les étapes depuis le site web, on peut remarquer les commandes qui commence à s'afficher dans notre interface, constituant ainsi notre scenario. Ce dernier dont le but est d'accéder au site et chercher dans les deals du jour les livres proposés par Amazon. La figure 2 affiche la mise en place des commandes générer par selenium suite aux actions de l'utilisateur. Chacune des commandes constitue une action enregistre par le serveur de selenium suite à une interaction avec un élément par exemple la commande :
Open : spécifie un URL ouvert dans le navigateur.
Click, clickAndWait : click sur un élément spécifié dans la page web

Chaque commande est représentée par son type, son target et sa valeur. Selenium permet au testeurs d'insérer ou modifier les commandes affichées.

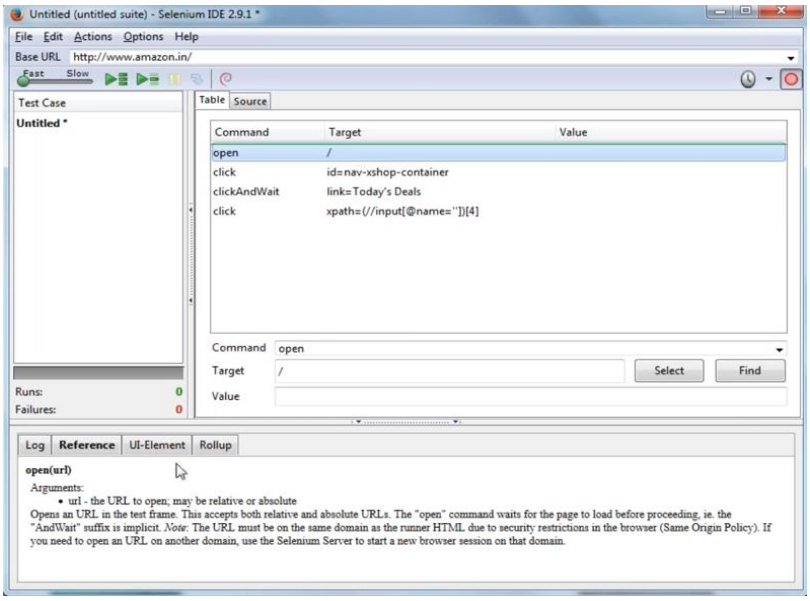

Figure 4: cas de test selenium

Dès que le cas de test est mis en place, on peut l'exporter sous différents langages de script générant ainsi le script approprié au cas de test. Les figures 3 et 4 montre successivement l'étape de d'exportation et le document du script généré.

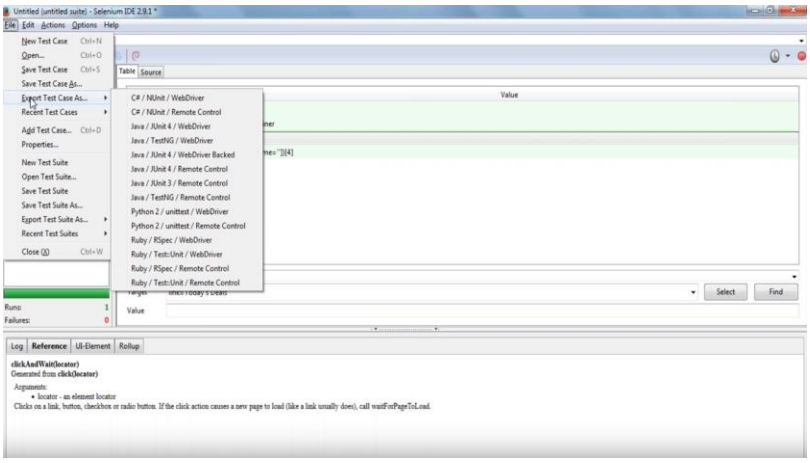

Figure 5: génération de script de cas de test

```
package com.example.tests;

import java.util.regex.Pattern;
import java.util.concurrent.TimeUnit;
import org.testng.annotations.*;
import static org.testng.Assert.*;
import org.openqa.selenium.*;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.support.ui.Select;

public class EdurekaAmazon {
  private WebDriver driver;
  private String baseUrl;
  private boolean acceptNextAlert = true;
  private StringBuffer verificationErrors = new StringBuffer();

  @BeforeClass(alwaysRun = true)
  public void setUp() throws Exception {
    driver = new FirefoxDriver();
    baseUrl = "http://www.amazon.in/";
    driver.manage().timeouts().implicitlyWait(30,
TimeUnit.SECONDS);
  }

  @Test
  public void testEdurekaAmazon() throws Exception {
    driver.get(baseUrl + "/");
    driver.findElement(By.id("nav-xshop-container")).click();
    driver.findElement(By.linkText("Today's Deals")).click();
    driver.findElement(By.xpath("(//input[@name=''])[4]")).click
```

Figure 6:resultat script généré

Katalon studio :

Un outil ou plutôt une plateforme de test automatisée qui se base sur les Framework open source selenium et Appium pour fournir des fonctionnalités afin de mettre en œuvre des solutions de test entièrement automatisées pour les applications web et mobiles. Le point fort de cette plate-forme et qui lui a fait gagner sa popularité c'est la facilite qui offre au testeurs pour commencer la création de leurs tests le plus rapidement possible réduisant ainsi l'effort et l'expertise requis pour apprendre et intégrer ces Framework.

Avantages :

- Outil open source en licence et en maintenance
- Regroupe les Framework et les fonctionnalités nécessaires pour la création et exécution des cas de tests
- Se base sur selenium Framework tout en supprimant le besoin de squilles de programmation

Inconvénients :

- Seulement les langages Java et Groovy sont supportés
- Les fonctionnalités continuent d'évoluer
- Sa communauté large augmente rapidement les solutions émergentes

Simulation :

Pour Katalon la simulation portera plutôt sur les tests d'une application mobile. Partant du même principe que pour sélénium, on a choisi une simple démonstration.
Avant de commencer il faut précise qu'on utilisera un simulateur « Android Emulator » afin de pouvoir créer notre cas de test depuis l'application mobile.
Après avoir créer le projet de démonstration, on choisit l'option test Cases comme affiché dans l'interface de Katalon dans la figure 1, puis il suffit d'uploader le projet de l'application mobile et choisir l'option « Spy web ».

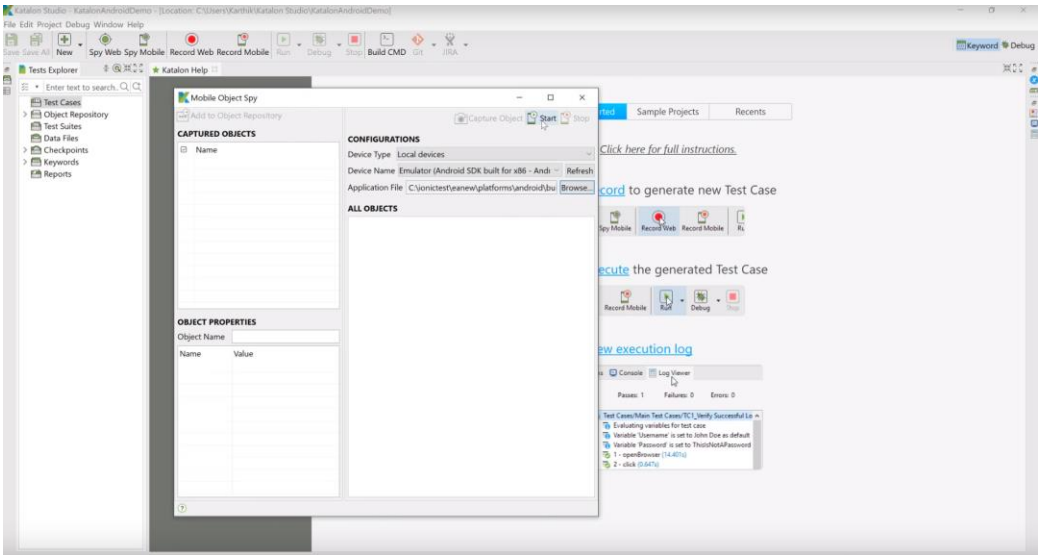

Figure 7: interface de création de projet Katalon

La figure suivante met en avant l'enregistrement des différentes actions qui ont été effectués depuis l'application, ces actions sont enregistrées sous forme de commandes comme c'est le cas pour selenium (voir simulation de l'outil précèdent). Après chaque évènement lancer depuis l'émulateur, l'interface offre la possibilité de définir l'action a réalisé et l'enregistre comme étant une action. Chaque élément appartient à un objet qui procède des propriétés.

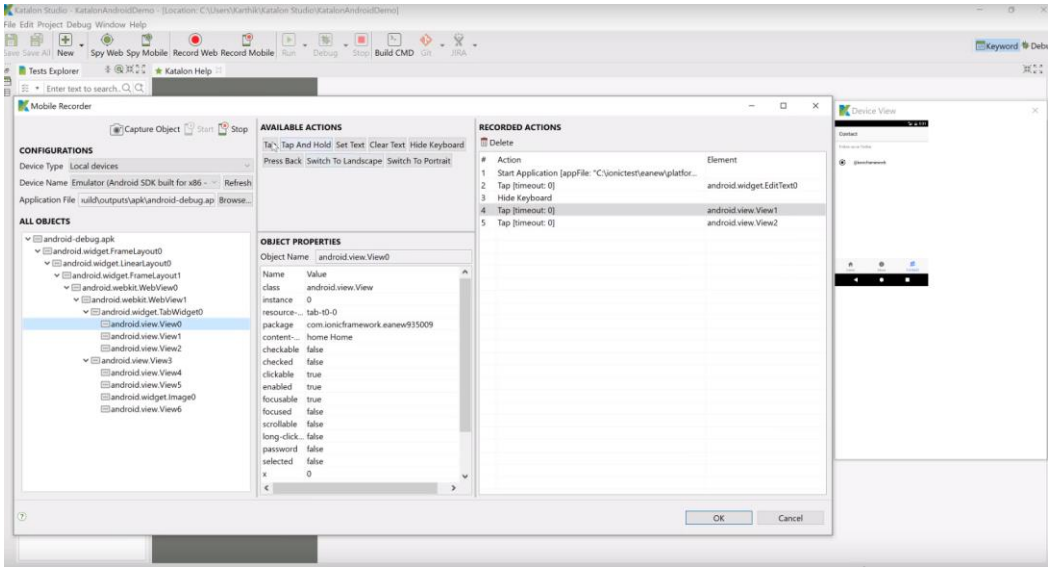

Figure 8 : génération des commandes

Une fois valide cette étape permet d'ajouter un nouveau cas de test, ainsi que les objets qui le constitue et qui apparaissent dans « object Repository » (voir figure). Une fois le cas de test créer on peut l'exécuter sous différente plateforme, dans notre cas, j'ai choisi de le faire sous Android, ce qui active le serveur Appium pour exécuter notre cas de test. Des informations sont fourni comme la date et temps d'exécution du cas de test, le temps d'exécution.

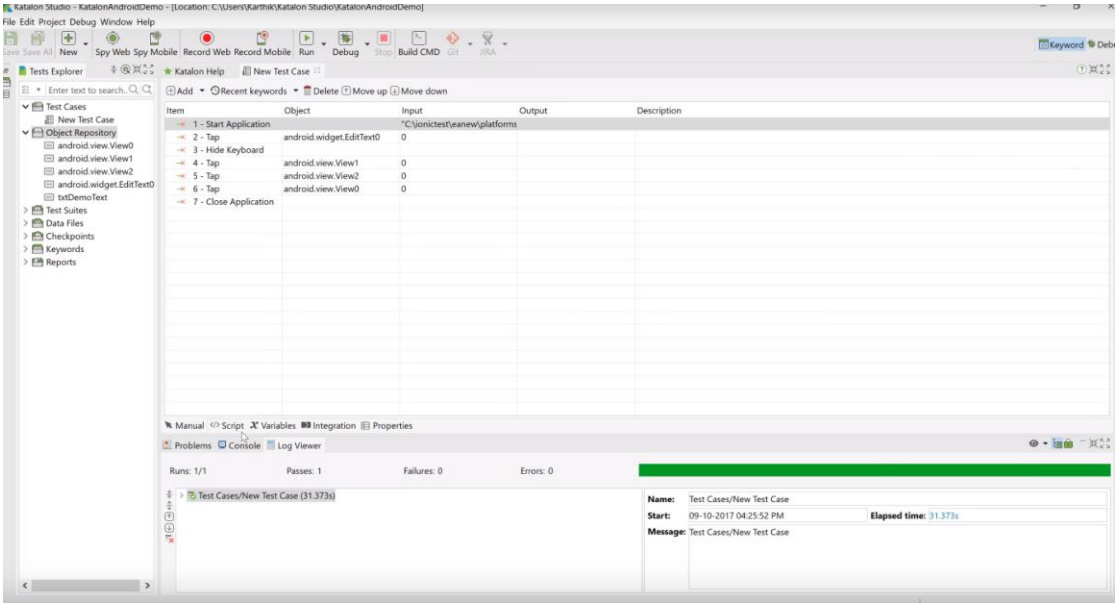

Figure 9: génération du cas de test

Un script est généré comportant toute les commandes du cas de test, avec la possibilité d'apporter des modifications ou des ajouts dans le cas de tests

```
import com.kms.katalon.core.testobject.ObjectRepository as ObjectRepository
import com.kms.katalon.core.testobject.TestObject as TestObject
import com.kms.katalon.core.webservice.keyword.WSBuiltInKeywords as WSBuiltInKeywords
import com.kms.katalon.core.webservice.keyword.WSBuiltInKeywords as WS
import com.kms.katalon.core.webui.keyword.WebUiBuiltInKeywords as WebUiBuiltInKeywords
import com.kms.katalon.core.webui.keyword.WebUiBuiltInKeywords as WebUI
import internal.GlobalVariable as GlobalVariable

Mobile.startApplication('C:\\ionictest\\eanew\\platforms\\android\\build\\outputs\\apk\\android-debug.apk', true)

Mobile.tap(findTestObject('android.widget.EditText0'), 0)

Mobile.hideKeyboard()

Mobile.tap(findTestObject('android.view.View1'), 0)

Mobile.tap(findTestObject('android.view.View2'), 0)

Mobile.tap(findTestObject('android.view.View0'), 0)

Mobile.closeApplication()
```

Figure 10: script cas de test

HP Unified Functional Testing :

Connu aussi sous Formely QuickTest Professional (QTP), il prend sa place dans ce classement puisque c'est peut-être l'outil commercial le plus populaire pour l'automatisation des tests fonctionnels. Il compte une variété de fonctionnalité qui permet facilement aux testeurs d'automatiser leurs tests pour la vérification automatique des fonctionnalités, quelle que soit la plateforme web, mobile ou desktop.

Avantages :

- Les fonctionnalités d'automatisation de tests intègre sont matures et compréhensibles
- Soutenus par une large communauté ainsi que d'un support dédié à l'utilisateur
- Pas besoin de connaissances avancées en programmation pour la création et l'exécution des tests

Inconvénients :

- Un cout élève pour la licence et la maintenance
- Plus de frais pour les modules additionnels
- Ne supporte que le langage VBScript

TestComplete :

TestComplete est aussi un outil appartenant à SMARTBEAR et qui doit sa popularité a sa capacité a supporté plusieurs langages comme Python, JScript, DelphiScript, C#, C++ et JavaScript et sa plate-forme intègre pour les applications web, desktop et mobile.

Avantages :

- Les fonctionnalités d'automatisation de tests intègre sont matures et compréhensibles
- Support de plusieurs langages
- Connaissance basic en programmation

Inconvénients :

- Comme pour UFT les couts sont élève pour la licence et la maintenance
- Plus de frais pour les modules additionnels

L'interprétation des résultats et la discussion

Comparaison des outils :

Le tableau suivant présente en résumé une comparaison faite par la communauté Katalon pour les outils les plus populaire cités précédemment. depuis les résultats présentent on constate que chacun des outils a ses propres caractéristiques fonctionnels qui les favorise, par exemple on a Katalon, UFT et TestComplet qui ont comme points commun la facilité d'utilisation et d'installation, rapidité dans la création des scripts ainsi que le non besoin de connaissances avancées des techniques de programmation, ce qui leurs donne un petit avantage sur Selenium, alors qu'à l'exception de Katalon ils sont payants et n'offre aucune analytique de tests. Tandis que Selenium est disponible en open source comme katalon et supporte plusieurs langages de Scripting ce qui est le même cas pour TestComplete et non pas pour les deux autres. [5]

Fonctionnalités	**Katalon Studio**	**Selenium**	**UFT**	**TestComplete**
Plateforme de développement de test	Cross-platforme	Cross-platforme	Windows	Windows
Application de test	Web et mobile apps	Web apps	Windows desktop, Web, mobile apps	Windows desktop, Web, mobile apps
Langages de script	Java/Groovy	Java, C#, Perl, Python, JavaScript, Ruby, PHP	VBScript	JavaScript, Python, VBScript, JScript, Delphi, C++ and C#
Compétences en	Non requis.	Niveau avancé	Non requis.	Non requis.

programmation	Recommandé pour les scripts de test avancés	requis	Recommandé pour les scripts de test avancés	Recommandé pour les scripts de test avancés
Courbe d'apprentissage	moyenne	élevé	moyenne	moyenne
Facilité d'installation et d'utilisation	Facile à installer et à exécuter	Require installing and integrating various tools	Facile à installer et à exécuter	Facile à installer et à exécuter
Temps de création de script	rapide	lent	rapide	rapide
Stockage et maintenance d'objets	Référentiel d'objets intégré, XPath, réidentification d'objet	XPath, UI Maps	Référentiel d'objets intégré, Détection et correction d'objets intelligents	Référentiel d'objets intégré, détection des objets communs
Test basé sur l'image	Support intégré	Installation de librairies additionnel requise	Support intégré, reconnaissance d'objet basée sur l'image	Support intégré
integrations DevOps/ALM	Plusieurs	Non (exige des bibliothèques supplémentaires)	Plusieurs	Plusieurs
Continuous integrations	Outils CI populaire(e.g. Jenkins, Teamcity)	Differents outils CI (e.g. Jenkins, Cruise Control)	Differents outils CI (e.g. Jenkins, HP Quality Center)	Differents outils CI (e.g. Jenkins, HP Quality Center)
Analytique de tests	Katalon Analytics	NA	NA	NA
support	Support par ticket, Communauté, staff dédié	Communauté Open source	Staff dédié, communauté	Staff dédié, communauté
Type de license	Freeware	Open source (Apache 2.0)	Proprietary	Proprietary
Coût	Gratuit	Gratuit	Frais de license et maintenance	Frais de license et maintenance

Tableau 2 : Comparaison des outils de tests

Conclusion :

Avec une progression rapide des logiciels et avec l'évolution des exigences de la clientèle, la complexité des applications augmente au fur et à mesure que les technologies évoluent, en plus du nombre de langage et des Framework les outils de tests se voient dans l'obligation de suivre le rythme de cette évolution. Bien qu'une évaluation des outils reste assez instructive, elle se voit limiter par la diversité des besoins.

Cette étude nous a permis de constate que les fournisseurs avant-gardiste de la technologie constitue une référence pour les outils de tests, aussi que les outils de type de test fonctionnel prennent plus de popularité puisqu'ils permettent de couvrir tous les niveaux de test. Bien que l'outil le plus populaire reste un open source la présence d'outils payant dans la liste montre que le prix de l'outil ne peut pas être considère comme un critère pour prévoir la popularité de l'outils.

D'un autre point de vue il est à tenir en compte qu'une recherche plus approfondie avec un contrôle plus avance des critères et des expérimentations dans l'état de l'art du domaine de la recherche dans le sujet loin des aspects économiques peuvent biaiser les résultats, ces derniers peuvent être plus pertinente.

Bibliographie :

[1] 829_WG - Std for Software Test Documentation Working Group. 2008. IEEE Std 829-2008 - IEEE Standard for Software and System Test Documentation
[2] Pierre Bourque, Richard E. 2018. IEEE Computer Society. SWEBOk Chapter 4: Software Testing.
[3] Alain April, Alain Abran. 2016. Améliorer la maintenance du logiciel, 2e edition chapitre 10.

[4] 2018. by International Software Test Institute. Software Testing Revealed 2ND edition
[5] communauté katalon. 2018. For Katalon. Comparison automated testing tools