Les tests d’acceptation sont un élément clé de la qualité des produits logiciels. Ils permettent de  
vérifier que le produit répond aux exigences du client et peut être utilisé de manière efficace. Dans un  
environnement agile, où la livraison rapide et la flexibilité sont des priorités, il est important de mettre  
en place une approche efficace pour les tests d’acceptation.  
L’objectif de cet article bibliographique est d’explorer les différents aspects de la mise en place des  
tests d’acceptation dans un environnement agile. Nous examinerons les avantages et les défis de l’in-  
tégration des tests d’acceptation dans une méthodologie agile, ainsi que les meilleures pratiques pour  
les mettre en œuvre avec succès. Nous explorerons également les outils et les technologies disponibles  
pour soutenir les tests d’acceptation agiles.  
Cet article bibliographique aidera les professionnels du développement logiciel qu’ils soient pro-  
grammeur,testeurs ou experts métiers qui cherchent à comprendre les tests et leurs raport avec les  
méthodes agiles.  
3

SABOUNI Ahmed 7 février 2023  
2 Les tests logiciels et leurs automatisations  
2.1 L’interêt des tests logiciels  
Les tests logiciels sont un processus systématique de vérification du logiciel pour s’assurer de son  
fonctionnement correct et de la conformité aux exigences spécifiées. Les tests logiciels sont importants  
pour garantir la qualité et la fiabilité du produit logiciel. Ils permettent de détecter les erreurs et les bugs  
avant la livraison, ce qui peut réduire les coûts liés à la correction des erreurs en production. Les tests  
logiciels peuvent également renforcer la confiance des clients et des utilisateurs finaux dans la qualité  
du produit, ce qui peut améliorer la réputation de l’entreprise et augmenter les chances de vente. Bien  
que les tests logiciels nécessitent un investissement en temps et en ressources, ils peuvent finalement  
permettre de réaliser des économies durant toute la durée de vie des produits.(UMAR, 2020)  
2.2 Classification des tests  
Les tests logiciels peuvent être classifiés en fonction de différents critères tels que la catégorie, le  
niveau, la technique et les types. Chacun de ces critères a ses propres avantages et inconvénients.  
— Catégories : Les tests peuvent être classés en tests dynamiques et tests statiques. Les tests dy-  
namiques peuvent être automatisés, tandis que les tests statiques sont généralement effectués  
manuellement. Les tests dynamiques ont l’avantage d’être plus rapides et plus fiables, mais  
peuvent être coûteux à mettre en place. Les tests statiques, quant à eux, sont moins coûteux,  
mais nécessitent une intervention humaine pour les effectuer et peuvent être plus longs et moins  
fiables.  
— Niveau : Les tests peuvent également être classés en fonction du niveau dans lequel ils sont  
effectués. Les tests de niveau unitaire sont effectués sur des composants individuels du logiciel.  
Les tests d’intégration sont effectués lorsque plusieurs composants du logiciel sont combinés et  
testés ensemble. Les tests de système sont effectués sur le système dans son intégralité. Enfin,  
les tests d’acceptation sont effectués pour vérifier si le produit répond aux exigences des utilisa-  
teurs. Chacun de ces niveaux de test a des avantages et des inconvénients spécifiques en termes  
de coût, de rapidité et de fiabilité.(TIAN, 2005)  
— Technique : Les tests peuvent également être classés en tests fonctionnels et non-fonctionnels.  
Les tests fonctionnels vérifient si le logiciel fonctionne correctement en répondant aux exi-  
gences fonctionnelles. Les tests non-fonctionnels, quant à eux, vérifient les aspects du logiciel  
tels que la performance, la sécurité et la qualité de l’expérience utilisateur. Les tests fonction-  
nels ont l’avantage de pouvoir être automatisés et de couvrir les exigences fonctionnelles, mais  
peuvent ne pas être suffisants pour couvrir les aspects non-fonctionnels du logiciel. Les tests  
4

SABOUNI Ahmed 7 février 2023  
non-fonctionnels, quant à eux, peuvent être plus difficiles à automatiser et à mesurer, mais ils  
fournissent une vue complète de la qualité du logiciel.  
— Types : Enfin, les tests peuvent également être classés selon leur type spécifique, tels que les tests  
de sécurité, de performance, d’utilisabilité, etc. Les tests de sécurité sont effectués pour vérifier  
la sécurité du produit et éviter les menaces potentielles, tandis que les tests de performance sont  
effectués pour vérifier la rapidité et l’efficacité du produit. Les tests d’utilisabilité sont axés sur  
l’expérience utilisateur et la facilité d’utilisation du produit.  
2.3 L’interêt des tests d’acceptation et l’importance de leur auto-  
matisation  
Les tests d’acceptation visent à évaluer le logiciel du point de vue de l’utilisateur et se concentrent  
sur les scénarios d’utilisation, les séquences, les modèles et les fréquences associées. Ils sont généra-  
lement plus adaptés aux logiciels lourds et leurs composants en tant qu’ensemble. Ils ont une relation  
étroite avec les clients et les utilisateurs et sont effectués dans les phases finales de test. L’environne-  
ment de test est similaire à celui utilisé par l’utilisateur final. Ces tests doivent étre décidé.  
tests d’acceptation  
Technique boite noir  
réalisé par testeurs et developpeurs  
Quand ? à la fin du projet  
Comment ? manuellement ou automatiquement (statique ou dynamique)  
Pourquoi ? pour vérifier si le produit répond aux exigences des utilisateurs  
Le test automatisé implique le développement de scripts de test à l’aide de langages de script tels que  
Python, JavaScript ou Tcl afin que les cas de test puissent être exécutés par des ordinateurs avec un  
minimum d’intervention humaine. La conception et le développement de tests peuvent être automatisés  
pour réduire l’effort humain et économiser de l’argent. Le logiciel d’automatisation peut également  
saisir des données de test dans le système en test, comparer les résultats attendus et réels et générer  
des rapports de test détaillés. Le test automatisé nécessite des investissements considérables en argent  
et en ressources. Les cycles de développement successifs nécessiteront l’exécution de la même suite  
de tests à plusieurs reprises. En utilisant un outil d’automatisation de tests, il est possible d’enregistrer  
cette suite de tests et de la rejouer au besoin. Une fois la suite de tests automatisée, aucune intervention  
humaine n’est nécessaire. Le but de l’automatisation est de réduire le nombre de cas de test à exécuter  
manuellement et non pas d’éliminer complètement les tests manuels. Les avantages du test automatisé  
sont la rapidité, l’efficacité coût, la répétabilité, la réutilisabilité, la programmabilité, la couverture  
complète et la fiabilité. Le test automatisé n’est pas toujours nécessaire, approprié ou rentable, mais  
5

SABOUNI Ahmed 7 février 2023  
une analyse peut aider à déterminer les bénéfices d’un test automatisé par rapport à un test manuel. La  
bonne gestion peut être obtenue en identifiant et en estimant les coûts et les avantages du test automatisé.  
6

SABOUNI Ahmed 7 février 2023  
3 Les tests agiles  
3.1 le testing agile et ses objectifs  
En général, les équipes de test cherchent à découvrir autant de bogues que possible dans le code  
développé, car les développeurs n’accordent que peu d’attention à la qualité. Les conséquences sont  
plus difficiles et coûteuses à corriger. Les équipes de test utilisent des outils pour documenter les cas  
de test et les bogues et peuvent externaliser la main-d’œuvre, mais cela ne résout pas les problèmes  
systémiques. Les activités de test intensives ralentissent la livraison et compromettent la qualité, sou-  
vent en raison de la pression de livraison tardive des projets et du manque de temps de test. Les défauts  
découverts à la dernière minute entraînent des retouches coûteuses et des problèmes plus graves si c’est  
un problème d’architecture ou de conception. Les équipes accumulent également une dette technique  
qui peut causer des retards futurs. La solution serait de déplacer les tests à des phases plus précoces du  
cycle de développement.  
3.2 Le développement piloté par les tests  
TDD, BDD et ATDD sont trois approches différentes pour développer des logiciels qui visent à  
garantir que le logiciel répond aux exigences et se comporte comme prévu.  
TDD : Un processus de développement de logiciels où les développeurs écrivent les tests en premier  
et écrivent ensuite du code pour faire passer les tests. L’accent est mis sur l’écriture de tests automatisés  
qui valident le comportement d’unités individuelles de code.  
BDD : Une extension du TDD qui se concentre sur l’écriture de tests à un niveau d’abstraction plus  
élevé. Il vise à décrire le comportement du système du point de vue de l’utilisateur final, ce qui rend les  
tests plus lisibles et compréhensibles pour les parties prenantes qui ne sont pas familières avec le code.  
7

SABOUNI Ahmed 7 février 2023  
ATDD : Semblable à BDD, l’ATDD est une approche de développement où les critères d’accepta-  
tion pour une histoire utilisateur sont définis avant le début du développement et des tests automatisés  
sont écrits pour valider que l’histoire a été correctement implémentée. L’accent est mis sur l’implica-  
tion des parties prenantes dans le processus de test pour garantir que les exigences ont été correctement  
implémentées.  
\*\*\*\*\*\*\*\*\*\* En résumé, TDD se concentre sur les tests unitaires, BDD se concentre sur les tests de  
comportement bout à bout et ATDD se concentre sur les tests d’acceptation avec l’implication des  
parties prenantes. \*\*\*\*\*\*\*\*\*\*\* Les scénarios BDD / ATDD sont souvent écrits en utilisant le format  
"Given, When, Then", qui décompose le scénario en trois parties distinctes : "Given" décrit le contexte  
de départ de la situation, "When" décrit l’action qui déclenche le comportement à tester, et "Then" décrit  
les résultats attendus suite à l’action décrite dans "When". Ce format permet de clarifier les attentes du  
comportement du système, de fournir une documentation claire et concise des tests, et peut être utilisé  
en conjonction avec des outils de test automatisé pour automatiser les tests et faciliter le processus de  
développement logiciel.  
8

SABOUNI Ahmed 7 février 2023  
3.3 les contrainte des tests agiles  
— Coût élevé : la mise en place d’une infrastructure de test automatisé peut être coûteuse en termes  
de temps et de ressources.  
— Temps de développement prolongé : l’automatisation des tests peut prendre beaucoup de temps  
et peut ralentir le processus de développement.  
— Maintenance difficile : les tests automatisés peuvent nécessiter une maintenance constante pour  
s’assurer qu’ils restent pertinents et précis, ce qui peut être coûteux en temps et en ressources.  
— Difficulté à tester des scénarios complexes : il peut être difficile de tester des scénarios com-  
plexes avec des tests automatisés, ce qui peut entraîner une couverture insuffisante.  
— Dépendance de la qualité du code : la qualité des tests dépend directement de la qualité du code,  
donc si le code n’est pas correctement écrit, les tests peuvent ne pas être fiables.  
— Dépendance des compétences techniques : les tests automatisés nécessitent des compétences  
techniques pour les écrire et les maintenir, ce qui peut limiter leur utilisation dans certaines  
équipes.  
Cependant, il convient de noter que ces inconvénients peuvent être minimisés en mettant en place une  
bonne stratégie de test et en choisissant les outils appropriés pour automatiser les tests. Les tests Agile  
peuvent également offrir de nombreux avantages, notamment une réduction des coûts, une amélioration  
de la qualité et une augmentation de la rapidité de déploiement.

.

├── dir1

│   ├── file11.ext

│   └── file12.ext

├── dir2

│   ├── file21.ext

│   ├── file22.ext

│   └── file23.ext

├── dir3

├── file\_in\_root.ext

└── README.md

├───src

│ ├───main

│ └───test

│ ├───java

│ │ ├───model

│ │ ├───Definition\_Etape

│ │ │ ├───DefFonctionnalité1.java

│ │ │ └───DefFonctionnalité2.java

│ │ └───autres

│ │

│ └───resources

│ ├───fonctionnalité1.feature

│ └───fonctionnalité2.feature

Les experts métiers les developpeur et les testeurs décident la stratégie le plan les scenarios… sur jira

On exporte les tests vers le dépôt du code

On éxécute les tests

On revoie les résultats vers jira

Dans un projet de test, les équipes de développement, de test et les experts métiers collaborent sur un outil de gestion de projet pour déterminer une stratégie de test. Chacun apporte son expertise en matière de développement dirigé par les tests (ATDD). Ils planifient les scénarios de test, les fonctionnalités à tester et les parties à automatiser. Une fois les fonctionnalités développées, les tests sont exportés vers le dépôt de code, où ils sont exécutés, et les résultats sont examinés et revus sur l'outil de gestion de projet. Cependant, on ne teste pas que les nouvelles fonctionnalité mais aussi ceux qui existaient déjà pour étre sur que les nouvelles fonctionnalité n’ont pas infécté celles qui existaitent déjà .dans ce cas nous parlons des tests de non regressions, Ces tests sont idéalement déjà développés mais peuvent nécessiter une rectification. Dans ce cas, on utilise un outil d'intégration continue pour assurer la qualité du logiciel.