GRENOBLE ECOLE DE MANAGEMENT

Grenoble INP ensimag

THÈSE PROFESSIONNELLE

INTÉGRATION DE L'INTÉLLIGENCE ARTIFICIELLE DANS LE PROCESSUS DE TEST ET AUTOMATISATION

Réalisée par:

Diane Feulefack

Encadrée par :

Mr Alain Rivet

Mastère Spécialisé Big Data

Promotion 2025

DEDICACES

À ma famille, en particulier à ma défunte maman **Feulefack Régine**, à mon papa **Gnimbouojouo Pierre** et à mon frère **Armel Mefo**. Vous êtes ma force et ma source d'inspiration.

REMERCIEMENTS

Mes vifs remerciements vont à l'endroit de:

- Mon tuteur de thèse, M. Alain Rivet, pour ses conseils précieux, son expertise ses remarques constructives et son accompagnement tout au long de ce projet ont été essentiels pour m'aider à approfondir mes recherches et à structurer mon travail.

- Mme Virginie Monvoisin notre professeur pour ces masterclass sur la rédaction de cette thèse.

- Mme Celine Roche Roussel pour son vaillant accompagnement tant administratif que accadémique.

- Mes professeurs et collègues de GEM et ENSIMAG, qui m'ont inspiré et soutenu durant mes études, en partageant leurs connaissances et leur passion pour l'innovation dans le champ de la data ainsi que de l'intelligence artificielle.

- Enfin, Je n'aurais pu mener ce projet à bien sans la patience et la présence constante de ma famille, dont le soutien m'a apporté sérénité et force à chaque étape.

ABREVIATIONS

AI Act – Artificial Intelligence Act
Règlement européen sur l'utilisation éthique, sécurisée et transparente de l'intelligence artificielle.

API – Interface de programmation d'applications
Mécanisme qui permet à différents logiciels ou systèmes d'échanger des données et de fonctionner ensemble en facilitant leur interaction.

AWS – Amazon Web Services
Ensemble de services cloud proposés par Amazon, offrant des solutions pour le stockage, le calcul, l'analyse de données et le déploiement d'applications dans une infrastructure hautement scalable et sécurisée.

CI/CD – Continuous Integration / Continuous Deployment
Intégration et déploiement continus: ensemble de pratiques DevOps visant à automatiser la construction, le test et la mise en production des logiciels.

CRM – Customer Relationship Management
Système de gestion permettant de centraliser les interactions avec les clients et d'optimiser la relation commerciale, le suivi et la fidélisation.

CX – Customer Experience
Perception globale qu'un client a d'une marque à travers l'ensemble de ses interactions, de la découverte du produit jusqu'au service après-vente.

DL – Deep Learning (Apprentissage profond)
Sous-domaine du machine learning qui exploite des réseaux de neurones à plusieurs couches pour traiter d'importants ensembles de données et identifier de manière autonome des modèles ou structures complexes.

DOM – Document Object Model
Modèle hiérarchique représentant la structure d'un document web, permettant aux programmes et scripts d'accéder, de modifier ou de tester les éléments d'une page.

ERP – Enterprise Resource Planning
Système intégré qui regroupe et automatise les principales fonctions d'une organisation telles que la comptabilité, la production, la gestion des stocks, la logistique et les ressources humaines au sein d'une plateforme unique.

FTP – File Transfer Protocol
Méthode standard permettant d'échanger des fichiers entre différents ordinateurs connectés à un même réseau.

GPT – Generative Pre-trained Transformer
Ensemble de modèles de traitement du langage créés par OpenAI, conçus pour produire et interpréter du texte en langage naturel.

HR / RH – Human Resources / Ressources Humaines
Département chargé de la gestion administrative, du recrutement et de la formation du personnel.

IA – Intelligence Artificielle
Branche de l'informatique qui cherche à simuler les capacités intellectuelles humaines telles que l'apprentissage, le raisonnement ou la perception au sein de systèmes numériques.

IoT – Internet des Objets
Réseau d'appareils physiques interconnectés via Internet, capables de collecter, échanger et transmettre des informations en continu afin d'automatiser ou d'optimiser diverses activités.

ISO/IEC – International Organization for Standardization / International Electrotechnical Commission
Institutions internationales chargées de définir et de promouvoir des standards en matière de qualité, de sécurité et de performance, en particulier dans les domaines des logiciels et des technologies de l'information.

JIRA – (outil Atlassian)
Logiciel de gestion de projets permettant de planifier, suivre et documenter les activités de développement ou de test.

JSON – JavaScript Object Notation
Format de données textuel simple et léger servant à échanger des informations entre différents systèmes.

LLM – Large Language Model
Modèle de traitement du langage naturel de grande capacité, reposant sur des réseaux neuronaux avancés tels que GPT ou BERT.

MLOps – Machine Learning Operations
Ensemble de pratiques associant le machine learning et les méthodes DevOps pour faciliter le déploiement, la surveillance et la mise à jour des modèles d'intelligence artificielle.

ML – Machine Learning
Sous-domaine de l'intelligence artificielle où les systèmes apprennent à partir des données afin d'améliorer leurs performances sans être programmés explicitement.

MTTR – Mean Time To Repair
Temps moyen de résolution: indicateur de performance mesurant la rapidité avec laquelle un système défaillant peut être restauré.

NLP – Natural Language Processing
Sous-domaine de l'intelligence artificielle qui vise à permettre aux ordinateurs d'analyser, de comprendre et de produire du langage humain, à l'écrit comme à l'oral.

OE – Original Equipment
Fabrication d'équipements d'origine: produits fabriqués pour être utilisés dans les chaînes de production de constructeurs automobiles.

PIM/DAM – Product Information Management / Digital Asset Management
Systèmes de gestion des informations produits et des contenus numériques associés (images, vidéos, documents techniques).

PLM – Product Lifecycle Management (Gestion du cycle de vie des produits)
Méthodologie intégrée visant à centraliser et piloter l'ensemble des informations et processus relatifs à la conception, à la fabrication et au suivi opérationnel d'un produit tout au long de son existence.

PWA – Progressive Web App
Application web progressive offrant une expérience similaire à celle d'une application native sur mobile.

QA – Quality Assurance
Assurance qualité: ensemble des activités visant à garantir la qualité de l'application tout au long de son cycle de mise en vie.

ROI – Return on Investment (Retour sur investissement)
Mesure financière servant à déterminer la rentabilité d'un projet ou d'un investissement en comparant les bénéfices générés aux ressources ou coûts engagés.

S&OP – Sales and Operations Planning
Processus de planification collaborative visant à aligner les prévisions de ventes avec les capacités de production et les ressources disponibles, afin d'assurer un équilibre durable entre la demande et l'offre.

SAP – Systems, Applications, and Products in Data Processing
ERP mondialement connu utilisé pour la gestion intégrée des processus d'entreprise.

SQL – Structured Query Language (Langage de requêtes structurées)
Langage informatique normalisé servant à concevoir, interroger, mettre à jour et administrer des bases de données relationnelles.

TDD – Test-Driven Development
Méthodologie de développement logiciel consistant à écrire d'abord les tests unitaires avant d'implémenter le code correspondant, afin d'assurer la qualité et la robustesse du logiciel.

TIA – Test Impact Analysis
Technique permettant de déterminer quels tests doivent être exécutés en fonction des changements récents dans le code.

UI – User Interface
Interface utilisateur: partie visible d'un logiciel avec laquelle interagit l'utilisateur.

UX – User Experience
Expérience utilisateur: qualité globale de l'expérience vécue par un utilisateur dans son interaction avec un produit numérique.

VM – Virtual Machine
Machine virtuelle: environnement logiciel simulant un ordinateur physique pour exécuter des applications isolées.

WMS – Warehouse Management System
Système de gestion d'entrepôt permettant d'optimiser la logistique, les flux de stockage et la traçabilité des produits.

XAI – Explainable Artificial Intelligence
Intelligence artificielle explicable: approche visant à rendre les décisions des modèles d'IA transparentes et compréhensibles.

XML – Extensible Markup Language (Langage de balisage extensible)
Format de balisage flexible permettant d'organiser, de stocker et de partager des données dans une structure compréhensible aussi bien par les humains que par les systèmes informatiques.

YAML – Yet Another Markup Language
Format de fichier lisible par l'humain, utilisé pour configurer des systèmes d'intégration continue ou d'orchestration de tests.

FIGURES

SOMMAIRE

INTRODUCTION

A) Contexte général et enjeux

Les logiciels sont devenus l'ossature de nos activités économiques, sociales et publiques. Dans ce contexte, tester un logiciel consiste à vérifier, par des méthodes organisées, qu'il se comporte comme prévu et qu'il répond à des exigences de qualité (fiabilité, performance, sécurité, utilisabilité). Historiquement, le test a longtemps été jugé coûteux, difficile à industrialiser et d'une efficacité parfois incertaine d'où une recherche continue de méthodes plus systématiques et plus automatisées. Ces limites sont documentées de longue date dans les travaux fondateurs de la discipline, qui rappellent le caractère souvent onéreux et imprévisible du test lorsqu'il est mal outillé [1] [2].

Parallèlement, les organisations ont profondément transformé leurs façons de produire du logiciel: cycles courts, livraison continue (CI/CD), équipes pluridisciplinaires (DevOps). Dans ces environnements rythmés, la rapidité d'exécution des tests et la capacité à détecter tôt les défauts deviennent déterminantes pour la performance opérationnelle. Des travaux empiriques sur la performance des équipes numériques montrent d'ailleurs que l'industrialisation des pratiques d'ingénierie (tests inclus) est associée à de meilleurs résultats business et techniques. [3]

L'automatisation des tests s'est imposée comme une première réponse à ce besoin d'efficacité. En permettant la réexécution rapide et systématique de suites de tests, elle a offert un gain substantiel en répétabilité, en traçabilité et en fiabilité. Cependant, cette automatisation dite "classique" atteint ses limites. Les scripts de test sont fragiles, sensibles à la moindre évolution de l'interface; la maintenance devient coûteuse; et la couverture fonctionnelle reste souvent partielle. De plus, le problème de "l'oracle" – comment décider automatiquement si le résultat d'un test est correct – demeure une question ouverte, rendant difficile une automatisation complète du jugement. [4]

C'est dans ce contexte qu'intervient l'intelligence artificielle (IA). Ses capacités d'apprentissage et d'adaptation ouvrent la voie à des tests plus intelligents: priorisation automatique des cas les plus critiques, génération dynamique de scénarios, détection d'anomalies visuelles ou comportementales, réparation automatique de scripts brisés. L'IA promet d'aller au-delà de la simple automatisation: elle permet d'apprendre à partir des données historiques de test et d'optimiser en continu les processus de validation. Cependant, cette promesse reste à nuancer: la dépendance aux données, la complexité

des modèles et la question de l'explicabilité imposent de nouvelles exigences méthodologiques et éthiques. Ces enjeux, à la croisée du génie logiciel et de l'intelligence artificielle, définissent l'espace dans lequel se définit cette recherche.

Origine de la recherche

L'idée initiale de cette recherche ne provient pas seulement d'une observation purement académique, mais d'un questionnement professionnel ancré dans la réalité du terrain. Au cours de mon parcours dans le domaine du numérique et de la transformation digitale, j'ai été confronté à des défis récurrents liés à la qualité des logiciels comme l'instabilité des tests automatisés, lenteur de validation, coûts de maintenance élevés, et difficulté à suivre la cadence imposée par les cycles CI/CD. Ces constats ont éveillé en moi un intérêt personnel pour les nouvelles approches de test, notamment celles mobilisant l'intelligence artificielle, dont la popularité ne cesse de croître dans les milieux techniques et managériaux.

C'est dans cette perspective qu'un questionnaire exploratoire a été diffusé via LinkedIn, afin de sonder la perception des professionnels sur le rôle de l'IA dans les tests logiciels. Ce réseau professionnel, riche en experts issus de secteurs variés (technologie, finance, industrie, télécommunications...), offrait un échantillon pertinent pour recueillir des opinions diversifiées. Le sondage a obtenu un taux de participation de 70 %, soit près de 50 réponses complètes sur une centaine d'invitations envoyées. Cette mobilisation a démontré l'intérêt croissant de la communauté IT pour ce sujet émergent.

Les résultats du questionnaire ont révélé des tendances particulièrement éclairantes. Une majorité des répondants (près de 80 %) considérait que l'IA représentait une opportunité majeure pour accélérer et fiabiliser les tests logiciels, mais une proportion non négligeable (près de 40 %) exprimait des réserves sur les risques de dépendance, de perte de contrôle et de manque d'explicabilité. Parmi les améliorations souhaitées, les participants ont relevés la nécessité d'une meilleure gestion des faux positifs et négatifs, d'une auditabilité accrue (traçabilité des décisions prises par l'IA), ainsi que d'une plus grande humanisation du processus, pour éviter que l'automatisation ne dégrade la compréhension du système testé. Plusieurs réponses insistaient sur l'importance de renforcer la transparence et la robustesse des algorithmes, des éléments aujourd'hui centraux dans le débat sur l'IA responsable.

Ces retours, riches et nuancés, ont été un élément déclencheur pour la présente recherche. Ils ont mis en évidence un écart entre les promesses technologiques et les perceptions réelles des praticiens, souvent partagés entre enthousiasme et prudence. L'analyse des réponses a permis d'identifier plusieurs questions-clés:

- Dans quelle mesure l'IA améliore-t-elle réellement la qualité et la rapidité des tests?
- Quels en sont les coûts cachés et les risques?
- Comment assurer la confiance et la responsabilité dans les décisions prises par un système intelligent?

Ces interrogations ont structuré la problématique de ce mémoire. Elles ont transformé une curiosité personnelle en une démarche scientifique visant à explorer, de manière empirique et critique, les conditions dans lesquelles l'IA crée de la valeur dans les processus de test. En cela, le questionnaire LinkedIn a joué le rôle d'un véritable "déclencheur de recherche" car il a permis de confronter une intuition professionnelle à une réalité collective et d'identifier les angles morts d'un sujet encore peu étudié sous un prisme organisationnel et humain.

Cette émergence du sujet s'inscrit dans une tendance plus large qui est la démocratisation rapide de l'IA dans les entreprises. Alors que des outils tels que ChatGPT, Copilot ou TestimAI se multiplient, les frontières entre testeur et algorithme deviennent floues. Les retours de terrain issus du questionnaire ont donc fourni une base empirique précieuse pour ancrer cette étude dans une réalité vécue, et non dans un simple discours technologique. C'est à partir de cette base que la recherche a pu se développer, combinant approche académique et réflexion issue de la pratique.

Pertinence du sujet et lien avec les transformations actuelles de l'ingénierie logicielle

L'adoption de solutions d'intelligence artificielle dans les activités de test logiciel ne constitue pas un phénomène isolé, mais s'inscrit pleinement dans la dynamique plus large de la transformation numérique et de l'évolution des pratiques d'ingénierie. Ces dix dernières années ont vu la naissance de nouveaux paradigmes: DevOps, intégration et déploiement continus (CI/CD), observabilité, infrastructure as code qui ont profondément modifié la manière dont les logiciels sont conçus, testés et déployés. Dans ces environnements, le test n'est plus une phase terminale du cycle de vie, mais un élément itératif et intégré au cœur même du processus de développement. L'IA apparaît ainsi comme une extension naturelle de cette logique d'automatisation et d'optimisation continue.

L'un des moteurs majeurs de cette transformation est la recherche d'efficacité opérationnelle. Les organisations modernes déploient des centaines, voire des milliers de mises à jour logicielles par an. Tester manuellement chaque modification est devenu impossible. L'automatisation a permis d'absorber une partie de cette charge, mais elle a aussi révélé de nouvelles contraintes: scripts fragiles, dette technique accumulée, manque d'adaptabilité. L'intelligence artificielle répond à ces limites en introduisant une couche d'apprentissage adaptatif: elle permet d'analyser les historiques d'exécution, d'identifier les tests les plus critiques, de détecter les défaillances récurrentes et d'optimiser la planification des campagnes de test (Pan et al., 2022). Ces avancées permettent de réduire le temps de feedback et d'améliorer la stabilité des pipelines CI/CD.

Cette évolution technique s'accompagne d'une mutation culturelle. Le mouvement DevOps, en favorisant la collaboration entre développeurs, testeurs et opérationnels, a transformé le test en outil de gouvernance de la qualité. Dans ce contexte, l'IA ne remplace pas le testeur, mais augmente ses capacités cognitives: elle aide à interpréter les anomalies, à prioriser

les correctifs et à contextualiser les incidents. Elle agit comme un partenaire intelligent capable de traiter des volumes de données massifs et d'en extraire des signaux pertinents.

La pertinence de ce sujet tient également à sa dimension stratégique. Dans des entreprises où le time-to-market est un indicateur clé, la capacité à livrer rapidement un produit de qualité devient un avantage concurrentiel majeur. Des acteurs comme Safran, Schneider Electric ou Sparco ont démontré que l'adoption d'outils d'analyse prédictive et de priorisation automatisée des tests se traduit par des gains mesurables en performance et en fiabilité. L'enjeu dépasse la simple technique car il touche à la résilience organisationnelle et à la capacité d'innovation.

On peut ainsi constater que l'étude de l'intelligence artificielle dans le domaine du test logiciel s'impose aujourd'hui comme un enjeu majeur, à la croisée de la recherche scientifique et des pratiques industrielles. Elle répond à une double exigence: celle d'accompagner la mutation technologique du génie logiciel moderne et celle d'en assurer la maîtrise méthodologique et humaine.

B) De l'automatisation à l'IA: une promesse d'efficacité sous conditions

L'intelligence artificielle, en particulier l'apprentissage automatique (machine learning), promettent d'aller au-delà de l'automatisation «classique». Concrètement, des techniques d'IA aident à:

- Prioriser les cas de test pour exécuter d'abord ceux qui ont le plus de chances de détecter des défauts,
- Générer automatiquement des données et des scénarios de test,
- Réparer des scripts cassés après une évolution du logiciel,
- Identifier des tests «flaky» (instables),
- Détecter des anomalies visuelles ou comportementales difficiles à formaliser en règles simples.

Des revues systématiques récentes recensent ces usages et en soulignent les bénéfices récurrents (réduction de l'effort manuel, amélioration de la couverture, meilleure détection de défauts), tout en rappelant des limites persistantes: outillage encore hétérogène, investissements initiaux, besoin de compétences spécialisées, et manque d'évaluations intégrées à l'échelle de l'entreprise. [5]

Il existe également une continuité naturelle entre les approches d'optimisation heuristique (par ex. la génération de jeux d'essai par recherche) et les approches d'IA contemporaines, qui exploitent des modèles pour explorer intelligemment l'espace des tests sous contraintes. Cette filiation méthodologique est documentée par les travaux sur le search-based testing.

C) Problématique

Dans un contexte où les entreprises souhaitent accélérer et fiabiliser leurs livraisons logicielles, l'IA apparaît comme un levier possible pour transformer les pratiques de test. Mais cette transformation n'est ni automatique ni sans coûts. La question centrale qui guide ce mémoire est donc:

Quels sont les avantages et les inconvénients de l'intégration de l'intelligence artificielle dans les processus de test et d'automatisation des tests logiciels?

Autrement dit, dans quelles conditions l'IA crée-t-elle de la valeur mesurable pour l'activité de test (efficacité, qualité, coûts, délais), et quels risques ou limites (techniques, organisationnels, éthiques) doit-on anticiper?

D) Intérêt académique et managérial

Sur le plan académique, Cette recherche s'inscrit dans la prolongation d'une étude antérieure de recherche qui interroge les fondements du test (définition des oracles, stratégies de génération, couverture) et explore les apports de l'IA pour automatiser des activités jusqu'ici manuelles. Il apporte une synthèse structurée des connaissances et discute des résultats empiriques disponibles, en soulignant les zones encore peu explorées (évaluations à grande échelle, intégration outillée «de bout en bout», gouvernance et explicabilité des modèles). [4] [5]

Sur le plan managérial, l'étude propose un cadre d'aide à la décision pour des directions IT et des responsables qualité: où déployer l'IA en priorité dans un portefeuille d'activités de test? quels prérequis (données, compétences, outillage, pilotage)? comment mesurer le retour sur investissement (KPI associés à la qualité et au flux)? Ces questions rejoignent les préoccupations des organisations engagées dans des démarches DevOps/CI-CD à forte cadence de livraison. [3]

E) Positionnement et périmètre

Afin de rester compréhensible par des lecteurs non spécialistes, ce mémoire adopte un vocabulaire simple et définit les termes techniques au fil du texte. Le périmètre couvre principalement:

- les tests fonctionnels (web/mobile/API) et les activités connexes (priorisation, sélection, génération de cas et de données, analyse de résultats, détection d'anomalies)
- les pratiques d'automatisation applicables dans des chaînes d'intégration et de déploiement continus
- les impacts organisationnels (compétences, processus, gouvernance, éthique)

Sont en revanche hors périmètre: la prédiction générique de défauts au niveau du code sans lien direct avec les activités de test traitées ici, et les considérations de R&D algorithmiques ne présentant pas d'enjeu opérationnel pour le testing.

F) Démarche méthodologique (aperçu)

Pour répondre à la problématique, la démarche combine:

1. Une revue de littérature académique structurée (articles de conférences et revues spécialisées en génie logiciel), qui sert de socle théorique et de cadre d'analyse. La conduite de cette revue s'inspire des lignes directrices reconnues pour les revues systématiques en ingénierie logicielle
2. Une étude qualitative multi-cas, fondée sur des entretiens semi-directifs et/ou des analyses documentaires d'entreprises ayant amorcé l'intégration de l'IA dans leurs tests. L'objectif n'est pas la généralisation statistique, mais une compréhension en profondeur des conditions de succès et des freins rencontrés, avec triangulation des sources lorsque c'est possible

Ce positionnement corrige explicitement l'ambiguïté relevée dans la version précédente: il ne s'agit pas rien que d'une étude quantitative, mais bien d'une analyse qualitative appuyée sur des cas réels et un ancrage académique solide.

G) Contributions attendues

Le mémoire propose:

- une carte des usages de l'IA dans les processus de test (activités, techniques, apports/risques) issue de la littérature et du terrain
- une grille d'évaluation pragmatique pour décider «où» et «comment» investir (pré-requis de données, coûts initiaux, compétences, métriques de succès)
- des recommandations managériales pour piloter un projet d'intégration (gouvernance, explicabilité, mesure de la valeur, conduite du changement)
- l'identification de limites et de pistes de recherche (outillage intégré, évaluations longitudinales, impacts sur la qualité globale)

H) Organisation du document

Ce document est structuré comme suit:

- L'Introduction qui pose le contexte, la problématique et la démarche
- Le Chapitre 1: Qui présente 4 la revues de littérature (définitions, analyses et critiques, état de l'art, débats…)
- Le Chapitre 2: Précise les objectifs et questions de la recherche

- Le Chapitre 3: Décrit la méthodologie de mon travail
- Le Chapitre 4: Expose les études de cas et interviews
- Le Chapitre 5: Propose une analyse des avantages et inconvénients
- Le Chapitre 6: Synthétise l'analyse et propose des recommandations
- Le chapitre 7: Elabore les limites de ma recherche ainsi que ses perspectives futures
- La Conclusion générale qui répond à la problématique posée, et propose des recommandations et pistes pour des recherches futures

CHAPITRE 1 – REVUE DE LITTÉRATURE

Dans un contexte de développement logiciel à cadence élevée (intégration et déploiement continus, méthodes agiles, DevOps), le test n'est plus une étape terminale, mais une fonction qui aide à prendre la décision tout au long du cycle de vie. L'automatisation traditionnelle a amélioré la répétabilité et la vitesse, mais elle atteint des limites bien connues: fragilité des scripts lors des changements d'interface, couverture insuffisante des scénarios plausibles, difficulté récurrente à statuer automatiquement sur le résultat attendu (le problème de l'oracle) Dans ce contexte, l'intelligence artificielle (IA) au sens large (apprentissage automatique, vision, traitement du langage naturel/LLM, apprentissage par renforcement) est perçue comme un levier d'efficience: prioriser les tests pertinents, générer des cas et des données, aider à décider (verdict), réparer des scripts, analyser rapidement des échecs à grande échelle. Cette promesse soulève toutefois des questions: niveau de preuve des résultats, conditions de succès (données, outillage, compétences), coûts et risques (explicabilité, biais, gouvernance).

Les revues suivantes visent à établir un état de l'art structuré sur l'IA appliquée aux processus de test et à l'automatisation. La question qui se poserait directement serait de s'avoir quel a été la stratégie documentaire utillisée afin de repérer ces revues?

Alors, conformément aux bonnes pratiques en génie logiciel, ma recherche s'appuie sur quatre plateformes: Google scholar, Scopus, ResearchGate et HAL sous cette stratégie:

- Types de documents visés: revues systématiques (SLR), surveys, systematic mappings, et, le cas échéant, revues multivocales incluant littérature industrielle lorsqu'elles suivent un protocole explicite.
- Période: 2004–2025, afin de couvrir les jalons «classiques» (search-based testing, métamorphique, oracles) et les avancées récentes (ML/LLM, vision, priorisation IA, flakiness).
- Langues: anglais majoritairement, ouvert au français via HAL.

Sous ces diffèrents sites j'ai trouvé des revues différentes mais qui plus ou moins convergent à la meme idée finale, Alors j'en ai recensé 4 que j'analyserai de facon indépendante l'unes de l'autres.

Mini-revue 1: *The Oracle Problem in Software Testing* [6]

Cette revue fondatrice clarifie le «problème de l'oracle»: dans de nombreux tests, il est difficile d'énoncer automatiquement un verdict fiable. L'article propose une cartographie des familles d'oracles et explique pourquoi ce nœud méthodologique limite l'automatisation, même dans des contextes outillés (CI/CD, DevOps).

Un oracle de test est le mécanisme qui permet d'affirmer qu'un résultat est correct (ou non): règle formelle, modèle, référence différentielle, jugement humain. Lorsque l'oracle est partiel, coûteux ou absent, l'automatisation se heurte à des faux positifs/négatifs, à des coûts de revue élevés et à une couverture trompeuse.

Barr et al. classent les oracles en cinq familles: humain, basé-spécification, différentiel, métamorphique, statistique/ML. Chaque famille implique des compromis entre confiance, coût et couverture. L'oracle basé-spécification offre la meilleure garantie quand la spécification est formelle et exécutable, mais elle est rarement complète. Le différentiel compare à une version ou à une implémentation de référence; utile en régression, il hérite cependant des défauts du référent. Le métamorphique vérifie des relations entre entrées et sorties lorsqu'on ne connaît pas la sortie exacte; il devient central pour des domaines où la vérité terrain est difficile à formaliser (vision, optimisation, ML). Les oracles statistiques/ML apprennent un comportement «normal» et signalent des anomalies, au prix d'une explicabilité parfois limitée.

Analyse et critique

- Forces: clarté conceptuelle, vocabulaire stabilisé, taxonomie actionnable; l'article sert de langage commun pour relier pratiques industrielles et travaux de recherche.
- Limites: focalisation principalement théorique, peu d'évaluations à grande échelle; publication pré-LLM et antérieure aux cadences CI/CD; absence de métriques normalisées pour comparer la «qualité» d'un oracle (précision, rappel, coût de revue, maintien dans le temps).

Tendances, lacunes et liens

- Tendances: essor d'oracles hybrides (règles + apprentissage), intérêt renouvelé pour le métamorphique, progression d'oracles visuels pour les IHM.
- Lacunes: manque de benchmarks publics, d'évaluations longitudinales et de mesures économiques (ROI, MTTR impacté). Lien naturel avec les revues sur la priorisation par ML (concentrer l'effort d'oracle) et sur la flakiness (qualité du signal).

Conclusion

Cette revue montre que l'oracle est le point critique de l'automatisation. Elle fournit une grille de choix contextuel et motive l'adoption d'indicateurs: précision/rappel du verdict, faux positifs, temps d'analyse, effort de revue et coût de maintenance. Elle ouvre un agenda

pragmatique: concevoir, mesurer et gouverner des oracles explicables, alimentés par des données traçables et intégrés à la CI/CD.

Mini-revue 2: Test Case Selection and Prioritization Using Machine Learning [7]

Cette revue étudie comment le machine learning améliore la sélection et la priorisation des tests de régression. En CI/CD, lancer toute la suite coûte temps et ressources; l'enjeu est de détecter plus tôt les défauts sous contrainte de budget. Les auteurs recensent les données exploitées, les modèles utilisés, les métriques d'évaluation (notamment APFD/APFDc) et les protocoles expérimentaux, puis discutent la validité des résultats rapportés par la littérature.

Synthèse des connaissances

- Données: historique d'exécutions (pass/fail, durée), couverture (lignes/fonctions), changements de code (fichiers touchés, churn, complexité), parfois métadonnées de criticité métier.
- Modèles: arbres de décision, random forests**,** gradient boosting**,** SVM, petits réseaux neuronaux; quelques travaux explorent apprentissage en ligne**,** bandits et renforcement.
- Résultats: gains fréquents sur le temps de feedback et l'APFD, surtout lorsque l'on impose un budget d'exécution. Les features liées aux changements récents et à l'historique de défaillances se révèlent particulièrement prédictives. Des études comparent l'IA à des heuristiques classiques (couverture, ordre chronologique, History-Based Test Prioritization) et montrent des améliorations, mais dépendantes du projet, de la qualité des données et de la stabilité des pipelines.

Tendances, lacunes et liens

Tendances: enrichissement des features dynamiques (graphes d'appels, dépendances), learning-to-rank, co-optimisation priorisation + budget, mise à jour en ligne des modèles.

Lacunes: benchmarks publics, protocoles comparatifs standardisés, indicateurs économiques (MTTR, débit), et explicabilité des choix.

Liens: articulation avec les oracles (qualité du verdict) et la flakiness (qualité du signal), qui conditionnent la fiabilité des données d'entraînement.

Conclusion

La SLR conclut que le ML accélère les retours de régression lorsque données fiables, baselines solides et métriques adaptées sont réunies. Pour ta thèse, elle fournit un cadre expérimental réutilisable (features, métriques, protocoles) et un agenda clair**:** cold-start**,** transférabilité inter-projets, coût total, explicabilité et preuves industrielles robustes**.**

Mini-revue 3: NLP-assisted Software Testing [8]

Cette revue cartographique s'intéresse à un thème émergent au croisement du traitement automatique du langage naturel (NLP) et du test logiciel. Alors que de nombreuses exigences logicielles sont rédigées en langage naturel, leur interprétation automatique reste un défi: ambiguïtés, vocabulaire hétérogène, manque de structuration. L'idée qui guide cette cartographie est que le NLP peut faciliter la transformation d'exigences textuelles en artefacts de test (cas, données, oracles). L'article s'inscrit donc à la fois dans la littérature sur l'automatisation du test et dans l'ingénierie des exigences, en fournissant une vue d'ensemble des approches et en identifiant les lacunes.

Les points clés sur lesquels s'applique cette littérature sont:

- NLP-assisted testing: utilisation de techniques NLP (analyse morpho-syntaxique, extraction d'entités, classification, transformation sémantique) pour produire ou améliorer les artefacts de test.
- Applications: génération de cas de test, extraction des informations à l'aide des exigences textuelles ou de rapports de bug, détection d'ambiguïtés dans la documentation, amélioration de la traçabilité.
- Problématique: comment fiabiliser et automatiser ce passage du texte à l'action de test, tout en réduisant le coût humain lié aux revues et à la formalisation manuelle.

Garousi et al. ont analysé environ 50 articles publiés entre 1998 et 2019, répartis en plusieurs catégories:

1. Génération de cas de test à l'aide d'exigences:
    - Méthodes basées sur la syntaxe (regex, analyse grammaticale).
    - Approches sémantiques (ontologies, modèles de domaine, word embeddings).
    - Outils capables de traduire des *user stories* ou des spécifications semi-structurées en scripts (par ex. en Gherkin).
2. Détection d'ambiguïtés et qualité des exigences:
    - Repérage de termes vagues («rapidement», «adéquat»).
    - Systèmes d'alerte pour la revue humaine.
3. Traçabilité et liens entre exigences et tests:
4. Analyse des rapports de bug:
    - NLP pour catégoriser les tickets, extraire des étapes de reproduction et proposer des scénarios de test récurrents.

De ces analyse des résultats clés en sont ressortis:

- Potentiel élevé de réduction d'effort et d'accélération de la génération de tests.
- Meilleure détection précoce des ambiguïtés → gain de qualité dès l'amont.

- Mais résultats très dépendants de la qualité linguistique des exigences et de la langue (anglais dominant).

Analyse critique

- Méthode: *systematic mapping study* rigoureux, avec protocole Kitchenham/PRISMA adapté, critères d'inclusion/exclusion explicites, classification thématique.
- Forces: première tentative large de cartographier ce domaine; transparence de la procédure; taxonomie utile pour chercheurs et praticiens; met en lumière les cas d'usage concrets (exigences, bug reports).
- Limites:
  - Corpus relativement réduit (≈50 études) pour plus de 20 ans.
  - Sur-représentation de petits prototypes académiques, peu d'études industrielles longitudinales.
  - Évaluations souvent qualitatives, sans métriques de couverture ni comparaisons robustes.

Tendances, lacunes et connexions

- Tendances: montée des techniques sémantiques, transition vers les word embeddings (systèmes connectés) à partir de 2015, premières incursions de modèles plus puissants (transformers) en fin de période.
- Lacunes: absence de benchmarks publics de spécifications, manque de reproductibilité, sous-exploration des langues autres que l'anglais, peu de validation industrielle.
- Connexions: cette revue complète les travaux sur la génération de tests et les oracles (revues 1 et 2). Le NLP apparaît comme une source de données d'entrée pour l'automatisation, mais doit être couplé à des mécanismes de validation humaine pour éviter les erreurs. Elle est aussi directement liée aux recherches récentes sur les LLM (ChatGPT, BERT), qui prolongent et amplifient ces premières approches.

Conclusion

Cette cartographie montre que le NLP est une piste prometteuse pour rendre le test plus accessible et moins coûteux en transformant les textes en artefacts exploitables. Elle souligne cependant les limites méthodologiques (manque de preuves quantitatives et de validations industrielles), la fragilité linguistique et le besoin d'hybrides (NLP + supervision humaine).

Cette revue est cruciale car elle offre un cadre historique (pré-LLM), une taxonomie claire des approches et un agenda d'amélioration (benchmarks, métriques, multilingue, industrialisation). Elle permet de situer les apports récents des LLM comme une réponse aux lacunes identifiées en 2020.

Mini-revue 4: *A Survey of Flaky Tests* [9]

Cette revue systématique (76 études) établit l'état de l'art le plus complet sur les flaky tests, c'est-à-dire des tests dont le verdict (pass/fail) varie alors que le code testé n'a pas changé. Les auteurs organisent la littérature en quatre volets: causes, coûts & conséquences, détection, atténuation & réparation, et rappellent que la flakiness constitue une menace à la validité de toute méthodologie qui suppose que l'issue d'un test dépend uniquement du code exercé. L'article sert de référence pour les équipes qui opèrent des pipelines CI/CD à grande échelle et cherchent à comprendre d'où vient l'instabilité, combien elle coûte, comment la détecter plus tôt et comment la réduire durablement.

Un test est dit flaky lorsqu'il passe puis échoue sans changement fonctionnel du système; les causes tiennent souvent à l'environnement (réseau, OS, horloge), à l'ordonnancement (concurrence, ordre d'exécution), à des attentes temporelles (timeouts, *sleep*), à des données non hermétiques, ou à des APIs non déterministes (hasard, horodatage). Des analyses empiriques fondatrices (sur des dizaines de projets open source) ont établi des taxonomies de ces causes et observé les stratégies de correction pratiquées par les développeurs.

Parry et al. mènent une revue systématique de la littérature, agrégée et structurée: définition d'un protocole de recherche, sélection/criblage des études, classification des résultats et synthèse critique. L'article organise la connaissance par catégories de causes, classes de techniques de détection (par exemple ré-exécutions, analyse statique, apprentissage automatique) et familles de remèdes (isolation, corrections de dépendances temporelles/environnementales, quarantaines, refactorings), en explicitant les menaces à la validité et les limites des approches.

Analyse critique

- Méthode: la revue synthétise 76 contributions et propose un vocabulaire commun (définition, typologie, critères de classification). Le protocole est transparent (sources, filtrage, schéma d'analyse), ce qui renforce la crédibilité de la cartographie. La structure «causes → coûts → détection → mitigation» est directement actionnable par les ingénieurs qualité et les équipes DevOps.
- Forces**:**
    - Clarté conceptuelle et langage commun.
    - Cartographie complète des facettes techniques et opérationnelles.
    - Forte utilité pratique (les quatre volets cadrent un plan d'action en CI/CD).

Limites: La revue hérite des limites du corpus: hétérogénéité des bancs d'essai et des métriques, reproductibilité parfois faible (artefacts non partagés), biais de plateforme/écosystème (Java dominant), peu d'évaluations longitudinales à très grande échelle. Sur le plan métrique, l'absence d'un noyau commun de KPI (ex. délai de feedback, MTTR, taux de faux positifs, coût de maintenance) rend difficiles les comparaisons «à armes

égales» entre techniques. Enfin, publiée en 2021, la revue est peu spécifique aux systèmes ML même si sa typologie des causes inclut naturellement le non-déterminisme qui touche ces systèmes.

Parry et al. n'analysent pas exclusivement les applications ML, mais leur cadre explique pourquoi des tests de pipelines d'apprentissage ou de modèles deviennent instables: stochasticité d'initialisation, parallélisme, bibliothèques numériques non déterministes, non-associativité des flottants, assertions statistiques (seuils d'accuracy) trop strictes. Des travaux focalisés ML montrent empiriquement que l'algorithme lui-même peut introduire du non-déterminisme et donc de la flakiness, et détaillent les correctifs typiques (fixer les seeds, ajuster les seuils, contrôler l'environnement, marquer/rejouer).

Tendances et lacunes

Tendances:

1. Prédiction assistée par apprentissage: apprendre, à partir de l'historique (pass/fail, durée, fréquence des changements, dépendances), quels tests sont à risque de flakiness, pour concentrer les efforts de stabilisation et éviter de «brûler» du temps de pipeline. Cette ligne s'articule avec les travaux de priorisation ML où l'objectif est l'optimisation du temps de feedback en tenant compte à la fois de la détectabilité des défauts et de la stabilité des tests.
2. Hygiène d'environnement: généralisation des environnements hermétiques (conteneurs, mocks, données figées), initialisation déterministe et politiques de rerun bornées avec quarantaine automatisée; ces pratiques s'outillent et se systématisent dans les chaînes CI/CD.
3. ML-spécifique: pour les tests de modèles, la recherche propose de calibrer les assertions pour équilibrer efficacité de détection et risque de flakiness; par exemple, FASER formalise la recherche d'un seuil optimal (tests non déterministes) comme un problème d'optimisation

Lacunes:

- Benchmarks publics et protocoles standardisés font défaut (notamment pour les projets ML), ce qui freine la comparaison rigoureuse des détecteurs et contre-mesures.
- Les mesures économiques (délai de feedback, taux de faux positifs, cadence de release) sont rarement rapportées de façon homogène, alors qu'elles conditionnent le ROI.
- L'explicabilité des prédicteurs (quels facteurs pèsent vraiment?) demeure limitée, ce qui freine l'adoption en entreprise.

Conclusion

De L'interaction IA ↔ données de test, nous pouvons dire que La flakiness pollue les labels (pass/fail). Un classifieur entraîné sur des verdicts bruités apprend le bruit; il faut donc nettoyer l'historique (débruitage, exclusions, quarantaines) et combiner ré-exécutions ciblées et tests différentiels pour désambiguïser les cas incertains avant apprentissage. Cette exigence découle logiquement du cadre de Parry et al., même si elle est détaillée par des travaux postérieurs focalisés ML.

La contribution de Parry et al. est structurante: elle fixe un cadre de référence qui transforme la flakiness d'un problème local en un problème scientifique et industriel avec causes typiques, coûts mesurables et familles de contre-mesures. Ce survey joue un rôle de charnière. D'un côté, il rappelle que le succès des outils IA (priorisation, génération, oracles, self-healing) dépend de la qualité du signal: sans maîtrise de la flakiness, l'IA accélère surtout la propagation du bruit. De l'autre, il suggère des solutions concrètes

- Instituer un tableau de bord de flakiness et une politique de rerun bornée
- Assainir les causes racines (ordre, temps, dépendances, données) en documentant les fixes
- Pour le ML, garantir des environnements déterministes quand c'est pertinent, et calibrer les assertions pour équilibrer efficacité de détection et risque d'instabilité. En somme, la valeur de l'IA en test ne se matérialise que dans un écosystème stabilisé exactement ce que la revue permet de concevoir et de piloter.

Synthèse générale

Les quatre revues éclairent, sous des angles complémentaires, ma question centrale: *quels sont les avantages et les inconvénients de l'IA dans les processus de test et d'automatisation?*

- Barr (revue 1) pose le décor: l'oracle de test, la capacité à décider automatiquement si un résultat est correct est le maillon faible de l'automatisation. Quand l'oracle est fragile ou absent, l'IA est tentante (oracles statistiques, vision, apprentissage de comportements), mais elle introduit des enjeux d'explicabilité, de faux positifs et de gouvernance. L'avantage promis (décision plus rapide, couverture accrue) est indissociable du risque (verdicts opaques, coût de calibration).
- Pan et al. (revue 2) montrent comment le machine learning améliore la sélection/priorisation en régression: utiliser l'historique (pass/fail, couverture, changements de code) pour exécuter *d'abord* les tests les plus informatifs. Les bénéfices sont clairs en CI/CD: retour plus rapide, APFD/APFDc améliorés, réduction du temps de pipeline. Mais les gains sont conditionnés par la qualité des données, le «cold-start» et la reproductibilité des études. Autrement dit, l'IA apporte de la valeur si je sais mesurer, nourrir et maintenir les modèles.
- Garousi et al. (revue 3) cartographient le NLP au service du test (exigences → scénarios/données). C'est le versant génératif: accélérer la conception de tests à partir de textes. L'intérêt est évident quand les backlogs sont volumineux, mais l'ambiguïté linguistique impose des garde-fous: revue humaine, dictionnaires/glossaires, et tests d'acceptation pour filtrer la sur-génération. Les LLM récents amplifient le potentiel… et le besoin de contrôles.
- Parry et al. (2021) synthétisent les tests "flaky" (instables). La flakiness dégrade la confiance et gaspille du temps; elle peut venir de l'environnement, des données, de l'ordre d'exécution ou de classifieurs/heuristiques mal calibrés. L'IA peut détecter/prédire la flakiness, mais peut aussi l'induire (décisions non déterministes, seuils mouvants, dépendance au contexte). L'apport réel de l'IA passe donc par une hygiène d'ingénierie (environnements hermétiques, données stables) et une gouvernance des modèles.

Au global, les avantages récurrents de l'IA (accélération des retours, meilleure focalisation, aide à la génération, tri intelligent d'échecs) s'échangent contre des coûts et risques: collecte et préparation de données, maintenance des modèles, explicabilité, reproductibilité, et flakiness potentiellement accrue si l'IA est mal intégrée. Ces revues justifient l'approche de ma thèse: évaluer l'IA non pas «en soi», mais dans un processus outillé, avec KPI de qualité (APFD/APFDc, MTTR, précision des verdicts, taux de flaky) et conditions de succès (données, intégration CI/CD, rôles, gouvernance).

En test logiciel, deux questions reviennent sans cesse: quoi tester en premier? et comment savoir automatiquement si c'est bon?

- Des chercheurs ont montré que des modèles d'IA peuvent choisir mieux l'ordre des tests et aller plus vite aux problèmes importants [10]
- D'autres ont montré qu'on peut générer des idées de tests en lisant les textes d'exigences (avec le traitement automatique du langage) [11]
- Mais il reste un gros nœud: décider automatiquement si un résultat est correct (l'oracle). On peut apprendre cette décision, mais alors on doit expliquer le verdict, contrôler les faux positifs, et mettre à jour le modèle.
- Enfin, certains tests sont instables (parfois verts, parfois rouges): c'est la flakiness. L'IA aide à les repérer, mais si elle est mal utilisée, elle peut ajouter de l'instabilité.

En clair: l'IA peut vraiment aider, à condition de mesurer ce qu'elle apporte, de laisser l'humain décider quand c'est sensible, et de soigner l'environnement (données propres, pipelines stables). Sans cela, on risque d'automatiser… des problèmes.

Analyse critique (forces, faiblesses)

Forces communes

Les quatre revues apportent une ossature conceptuelle solide:

1. Un vocabulaire stabilisé autour de l'oracle (Barr).
2. Un cadre expérimental et des métriques propres au test (Pan: APFD/APFDc, temps de feedback).
3. Une cartographie utile des usages NLP → tests (Garousi)
4. Une taxonomie des causes/coûts de la flakiness et des pistes de détection/mitigation (Parry).

Ensemble, elles tracent un chemin pragmatique pour intégrer l'IA: prioriser intelligemment, générer là où le texte est net, sécuriser l'oracle, combattre la flakiness.

Limites récurrentes

- Reproductibilité: peu de jeux de données publics et de pipelines partagés; comparer les approches est difficile.
- Externalité: plusieurs résultats sont obtenus sur des bancs académiques ou des projets de petite taille; la transférabilité à de grands SI reste à démontrer.
- Économie du test: rares sont les travaux qui chiffrent coûts et ROI (collecte de données, maintenance des modèles, dette technique).
- Explicabilité & gouvernance: les revues soulignent l'enjeu, mais la documentation et l'audit des modèles restent peu couverts.

- Pré-LLM pour Barr et partiellement pour Garousi: l'essor des LLM nécessite des mises à jour (gains réels vs hallucinations, garde-fous).

En résumé critique, Les quatre revues convergent: l'IA est un accélérateur si et seulement si je bétonne le processus autour d'elle. Le cœur est l'oracle (fiable et explicable), le levier est la priorisation (mesurée), le multiplicateur est le NLP/LLM (sous contrôle), et le frein majeur est la flakiness (à combattre en amont). Ma thèse s'attachera à mesurer ces échanges gains-risques dans des contextes réels, afin d'indiquer où l'IA crée de la valeur dans le test et où elle n'en crée pas.

CHAPITRE 2 – OBJECTIFS ET QUESTIONS DE LA RECHERCHE

La recherche sur l'intégration de l'intelligence artificielle au sein des processus de test logiciel est un domaine passionnant et en pleine expansion. Tester un logiciel revient très simplement, à vérifier qu'il fait bien ce pour quoi il a été conçu et qu'il ne casse rien d'important lorsqu'on le fait évoluer. Aujourd'hui, les applications changent très vite, nous avons des nouvelles fonctions, correctifs, mises à jour fréquentes. Dans ce rythme soutenu, les équipes n'ont pas toujours le temps ni les moyens d'exécuter tous les tests à la main. L'intelligence artificielle apparaît alors comme une promesse car elle permet le gain du temps, repérer plus vite les problèmes, automatiser des tâches répétitives. Autrement dit, l'IA est en train de redéfinir les règles, permettant aux équipes de développement d'aller plus vite et d'assurer une qualité plus élevée des logiciels. Mais cette promesse s'accompagne de questions: où l'IA apporte-t-elle vraiment de la valeur? quels risques introduit-elle? quelles conditions faut-il réunir pour que cela fonctionne dans la pratique?

Cette section expose les objectifs de ma thèse, en détaillant les points clés à étudier afin de comprendre en profondeur les avantages et les limites de l'IA dans les tests logiciels. Elle fixe aussi une ligne simple qui est celle de regarder l'IA dans le test avec lucidité, ni enthousiasme naïf, ni rejet a priori. Des questions claires, des mesures compréhensibles, des retours concrets. L'ambition est d'offrir, à des lecteurs non spécialistes, les clés pour décider où l'IA aide vraiment, à quel prix, et avec quelles précautions.

Cette recherche n' éssayera pas par contre à prouver que l'IA est "magique". Il s'agit au contraire d'éclairer les décisions où l'IA vaut la peine, où elle en vaut moins, et comment l'adopter avec prudence. Les résultats viseront la sobriété: faire mieux avec ce que l'on a déjà (données, outils, compétences), plutôt que d'imaginer des transformations lourdes et coûteuses.

Objectif principal

L'objectif général est d'évaluer de manière égale Les aspects positifs et négatifs de l'utilisation de l'intelligence artificielle dans les processus de test et l'automatisation des tests. En d'autres termes explorer comment l'intelligence artificielle impacte les tests logiciels, en particulier en ce qui concerne son efficacité, sa fiabilité, et les défis d'intégration dans des environnements réels. À travers cette étude, j'aspire à comprendre si l'IA tient réellement ses promesses et répond aux attentes croissantes des entreprises et des développeurs qui recherchent des méthodes de test plus intelligentes et plus rapides.

L'IA est souvent présentée comme la solution ultime pour améliorer la qualité des logiciels. Mais en réalité, qu'en est-il? Ce projet vise donc à aller au-delà des hypothèses et des idées reçues pour évaluer objectivement ce que l'IA peut apporter – et où elle atteint ses limites. L'ambition de cette thèse est de démêler les aspects théoriques des réalités pratiques, afin de fournir aux entreprises une vue complète et concrète sur les impacts de l'IA dans leurs processus de test.

Objectifs spécifiques

Pour répondre à cet objectif principal, des objectifs spécifiques sont définis. Ces objectifs guideront chaque étape Depuis la phase de recherche et de collecte des données jusqu'à l'interprétation des résultats.

1. **Analyser l'influence de l'IA sur l'efficacité des processus de test logiciel**

Aujourd'hui, l'une des promesses les plus souvent avancées concernant l'IA dans les tests logiciels est sa capacité à améliorer l'efficacité globale des processus. Cet objectif cherche à quantifier cet impact en explorant si l'IA permet effectivement de réduire le temps de test et d'agrandir la couverture, tout en maintenant une qualité optimale. Il s'agit de comprendre si les gains d'efficacité sont significatifs, et dans quelle mesure ils varient selon les secteurs d'activité ou le type d'application testée.

2. **Évaluer les avantages spécifiques de l'IA dans l'automatisation des tests**

Un autre objectif essentiel est de décortiquer les bénéfices spécifiques que l'IA peut apporter dans l'automatisation des tests. Beaucoup d'entreprises voient l'IA comme une solution pour automatiser des tâches répétitives, mais il est crucial de déterminer si cette automatisation est aussi performante qu'elle le prétend. L'objectif est de voir comment l'IA transforme l'automatisation, de la génération de cas de test à la détection proactive des anomalies, et d'identifier les domaines où elle excelle par rapport aux méthodes traditionnelles.

3. **Examiner les défis et les limitations liés à l'intégration de l'IA dans les tests logiciels**

Bien que l'IA offre des avantages indéniables, elle n'est pas sans ses propres défis. La mise sur pieds de l'IA dans les tests logiciels exige une infrastructure robuste, des compétences techniques spécialisées, et souvent un budget conséquent. Ce troisième objectif vise à identifier les obstacles majeurs auxquels les entreprises doivent faire face lorsqu'elles intègrent l'IA dans leurs processus de test. Il s'agit d'examiner les difficultés techniques, organisationnelles et financières, et d'analyser comment ces défis peuvent être surmontés.

4. **Etudier les perspectives futures et l'impact potentiel de cette technologie sur le développement des tests logiciels**

Enfin, l'un des points les plus fascinants de cette recherche est de se pencher sur l'avenir de l'IA dans les tests logiciels. Quelles seront les évolutions technologiques et organisationnelles qui transformeront encore plus les tests dans les années à venir? Cet objectif a pour but d'explorer les innovations futures, en posant les bases pour des recherches supplémentaires. Il permet de réfléchir aux impacts à long terme et de comprendre comment les entreprises pourraient se préparer pour l'avenir, en adaptant leurs méthodes de test pour tirer le meilleur parti des avancées en IA.

Questions de recherche

Pour répondre à ces objectifs, plusieurs questions de recherche guideront cette thèse. Ces questions ont été soigneusement formulées pour structurer l'analyse et fournir des réponses claires et précises aux aspects clés de l'intégration de l'IA dans les tests logiciels.

- L'IA améliore-t-elle significativement l'efficacité des processus de test logiciel? Cette question vise à comprendre si l'IA répond vraiment aux attentes en matière de réduction du temps et d'optimisation des ressources pour les tests logiciels.

- Quels sont les principaux avantages de l'IA dans l'automatisation des tests logiciels? En répondant à cette question, cette recherche pourra clarifier les bénéfices spécifiques de l'automatisation par l'IA, en comparant notamment la performance des processus IA aux méthodes de test traditionnelles.

- Quels défis et limites rencontrent les entreprises dans l'intégration de l'IA dans les tests?
  Cette question est cruciale pour saisir les difficultés techniques, financières, et humaines qui peuvent freiner l'adoption de l'IA dans les tests logiciels. Elle permettra d'apporter des pistes de solutions aux entreprises qui envisagent cette transformation.

- Quelles perspectives d'avenir l'IA apporte-t-elle aux tests logiciels, et quelles implications cela a-t-il pour les entreprises?
  Cette question porte un regard prospectif sur les innovations potentielles et les transformations attendues dans les prochaines années, permettant aux entreprises d'anticiper les évolutions et de préparer des stratégies à long terme.

CHAPITRE 3: METHODOLOGIE

Mon approche méthodologique repose sur un mélange de méthodes qualitatives et quantitatives, en d'autres termes une approche qualitative enrichie de données quantitatives descriptives. Cette approche est particulièrement adaptée pour examiner les différents aspets de l'intégration de l'IA dans les tests logiciels, car elle permet d'obtenir une compréhension approfondie des points de vue des professionnels, tout en appuyant ces perspectives par des données empiriques.

- Approche qualitative: Cette partie de la recherche prends son appuie sur un questionnaire diffusé auprès de professionnels via LinkedIn. J'ai choisi LinkedIn car c'est une plateforme où des experts de divers secteurs partagent leurs connaissances et leurs expériences. Ce réseau m' a permis de recueillir des réponses variées et d'atteindre un public de professionnels, qui offrent un regard pratique sur l'IA dans les tests logiciels.
- Approche quantitative: Pour enrichir les données qualitatives, une série d'études de cas a été réalisée auprès de 4 entreprises dont Sparco S.p.a, Opencell, Schneider, Safran qui ont intégré l'IA dans leurs processus de test. Ces études de cas permettent de comprendre l'impact de l'IA dans des contextes réels et variés, en analysant des indicateurs comme le temps de test, la couverture des tests, et coûts.

Collecte des données

Le processus de collecte des données s'est effectué en deux phases distinctes: la diffusion d'un questionnaire en ligne et la réalisation d'études de cas d'entreprises. Ces deux étapes complémentaires permettent d'obtenir une image complète des avantages et des défis de l'IA dans les tests logiciels.

Etude de cas multiples, comparaisons “avant vs après”

Je conduirai une étude de cas multiple (4 organisations), chaque cas étant une entreprise ayant effectivement intégré des techniques d’IA dans au moins une activité de test (ou pas forcément un test). Pour chaque cas, j’effectue une comparaison diachronique selon ces deux aspects:

- Période “Avant IA”: fenêtre de 6 à 12 mois immédiatement précédant le déploiement significatif d’IA dans le test (ou pas forcément).
- Période “Après IA”: fenêtre de 6 à 12 mois suivant la stabilisation du déploiement (après ~2–3 mois de rodage), afin d’éviter les effets de démarrage ou des intéruptions.

J’adopte cette logique avant/après car elle est un classique en génie logiciel lorsqu’on évalue des pratiques techniques (par exemple DevOps) au regard d’indicateurs de flux et de qualité; j’y adjoins une lecture qualitative des conditions d’adoption (données, outillage, compétences) et des effets collatéraux (coûts, gouvernance, acceptation).

Questionnaire diffusé via Linkedin

Pour recueillir des informations variées et représentatives, un formulaire de questions a été diffusé via LinkedIn, visant à toucher un large chapms de professionnels des secteurs technologique, bancaire, automobile et autres. Voici comment s'est déroulé le processus de collecte:

- Public ciblé: Le questionnaire visait des profils variés (développeurs, ingénieurs de tests, chefs de projet, managers techniques) pour obtenir une diversité de points de vue. L’objectif était de recueillir des avis provenant de différentes fonctions afin de capter les nuances de l’impact de l’IA selon les rôles.
- Structure du questionnaire: Le questionnaire a été structuré en plusieurs sections pour aborder des thèmes spécifiques: efficacité et productivité, précision et qualité des tests, défis et limitations, ainsi que les perspectives d’avenir. Chaque section comprenait des questions fermées (choix multiples) et des questions ouvertes pour encourager les participants à détailler leurs réponses. Cela a permis d’obtenir à la fois des données quantitatives et des insights qualitatifs.
- Statistiques de distribution: Environ 100 invitations ont été envoyées sur LinkedIn, et le taux de réponse a été de 70%, avec un total de 50 réponses complètes. Ces chiffres offrent une bonne base pour une analyse représentative, tout en permettant une diversité de secteurs et de perspectives.
- Exemple de questions posées:
    - Connaissez-vous l’IA? l’avez-vous déjà utilisé?
    - Percevez-vous des bénéfices/risques de son usage?

- L’IA a-t-elle amélioré la précision dans la détection des anomalies et des bugs?
- Quels sont les principaux défis que vous rencontrés lors de l'implémentation de l'IA dans les tests logiciels?
- Quels critères de confiance (explicabilité, responsabilité) vous lui accordez?
- Quelles améliorations souhaiteriez-vous voir dans l’utilisation de l’IA pour les tests logiciels?

Synthèse des résultats

Les résultats de cette recherche seront présentés selon une logique intégrée, combinant les données issues du questionnaire et celles recueillies lors des études de cas. Ces résultats obtenus contribueront directement à répondre à la problématique de la recherche: déterminer les avantages et les limites associées à l’intégration de l’intelligence artificielle dans les processus de test logiciel.

CHAPITRE 4: ÉTUDE DE CAS ET INTERVIEWS

ÉTUDE DE CAS

Le but principal de ce chapitre est de d'élaborer et d'analyser des études de cas concrets d'entreprises ayant amorcé L'incorporation de l'intelligence artificielle au sein de leurs processus de test logiciel. Ces cas permettent d'illustrer, au-delà de la théorie et des tendances issues de la littérature, les réalités pratiques de la mise en place de l'IA dans des contextes organisationnels et sectoriels différents.

La méthodologie adoptée repose sur une logique comparative «avant/après», déjà présentée au chapitre précédent. Chaque étude de cas est structurée autour d'une même grille d'analyse comprenant des indicateurs d'efficacité (temps de test, délai de feedback), de qualité (taux de bugs détectés, couverture), de coûts (ressources, outillage), et d'aspects organisationnels (compétences, gouvernance, adoption par les équipes). Cette approche diachronique permet de mesurer l'apport concret de l'IA et d'identifier les éventuelles limites rencontrées.

Les entreprises retenues représentent une diversité de secteurs et de contraintes:

- **Sparco S.p.a**, acteur industriel du secteur automobile et sportif, engagé dans des projets de transformation numérique et d'optimisation de ses processus qualité.
- **Opencell**, entreprise orientée vers le logiciel SaaS, caractérisée par des cycles de livraison rapides et une forte exigence en automatisation.
- **Schneider**, groupe international spécialisé dans l'énergie et le digital, doté d'une culture avancée de la donnée et du cloud, favorable à l'expérimentation de solutions IA.
- **Safran**, acteur majeur de l'aéronautique, où la sécurité et la fiabilité sont critiques, offrant un terrain d'analyse pertinent sur les enjeux de gouvernance et d'explicabilité des modèles.

L'analyse de ces quatre cas a pour objectif non seulement de dégager les bénéfices observés (accélération des tests, réduction des coûts, amélioration de la qualité), mais aussi de mettre en lumière les défis persistants (complexité de mise en place, compétences requises, résistance au changement). Au-delà des constats individuels, une synthèse transversale viendra souligner les régularités et différences, afin de tirer des enseignements généralisables sur les conditions de succès de l'intégration de l'IA dans les tests logiciels.

Cas 1: Safran Landing Systems

1. Contexte général et organisation

Safran est un groupe industriel de haute technologie spécialisé dans la propulsion aéronautique, l'aéronautique, la défense et le spatial. Avec plus de 92 000 employés et un chiffre d'affaires dépassant 23 milliards d'euros, le groupe est un acteur clé de la filière aéronautique mondiale.

Safran développe aussi des capacités très avancées en intelligence artificielle, via sa filiale Safran.AI créée à partir de l'acquisition de Preligens: son offre de solutions en deep learning, d'algorithmes pour la défense, la reconnaissance d'images, la géospatial intelligence, etc., renforce ses compétences [12] [13].

Ses filiales principales: S.Aircraft Engines, S.Landing Systems, S.Electronics & Defense et S.Nacelles, conçoivent des systèmes complexes où la fiabilité logicielle est critique pour la sécurité des vols.

Les logiciels embarqués développés par Safran pilotent des moteurs, calculateurs de vol, trains d'atterrissage, systèmes inertiels ou encore des modules de navigation satellitaire. Ces programmes doivent satisfaire à des normes strictes, telles que DO-178C, ARP4754A et DO-330, qui imposent des processus de test rigoureux, traçables et vérifiables.

Dans ce contexte, Safran a entrepris depuis 2020 une démarche structurée l'incorporation de l'intelligence artificielle dans les processus de validation et de test logiciel, notamment dans le cadre de ses programmes de R&D « Smart Testing » et « IA4Test », en collaboration avec Airbus, Thales et Capgemini.

L'objectif est double: accroître la productivité des équipes de test tout en renforçant la fiabilité et la traçabilité des validations logicielles critiques. L'IA intervient aujourd'hui dans plusieurs segments: génération automatique de cas de test, priorisation des campagnes, détection d'anomalies dans les journaux de vol et simulation des environnements complexes via des modèles prédictifs.

2. Situation avant l'introduction de l'IA dans les tests

Avant 2020, le processus de test chez Safran reposait sur une organisation très structurée, mais largement manuelle et séquentielle, en conformité avec les référentiels DO-178C:

- Conception manuelle des tests à partir des spécifications basées sur les exigences (Requirements-Based Testing).
- Utilisation de scripts Python, C et Ada pour les tests unitaires sur calculateurs embarqués.

- Vérification et validation (V&V**)** exécutées dans des environnements simulés (bancs HIL/SIL) nécessitant d'importants efforts de maintenance.
- Revue humaine des rapports de tests, avec une détection souvent tardive des anomalies complexes issues d'interactions entre logiciels et matériel.
- Durée des cycles de test longue: jusqu'à 90 jours pour un moteur complet, notamment à cause des exigences de traçabilité et de certification.

Cette approche, bien que robuste, présentait plusieurs limites:

- Faible réutilisation des données d'essais précédents (tests redondants).
- Difficulté à identifier les tests inefficaces ou à faible couverture.
- Coût élevé des campagnes d'analyse et de validation.
- Manque d'outils de visualisation et d'aide à la décision pour interpréter les milliers de logs issus des bancs d'essai.

Safran constatait ainsi un écart croissant entre les besoins de productivité imposés par les cycles agiles et les contraintes de sûreté de fonctionnement imposées par l'aéronautique. C'est dans ce contexte qu'a émergé le programme Smart Testing, soutenu par Safran Tech et le programme européen CleanSky 2.

3. Mise en place et intégration de l'IA

Safran a mis en œuvre plusieurs initiatives pour améliorer ses processus de test à travers l'IA, parmi lesquelles:

a) Automatisation intelligente et priorisation des tests

La première étape du programme Smart Testing a consisté à intégrer des modèles de machine learning dans les environnements de tests existants pour analyser les historiques d'exécution.

Ces modèles, développés en Python (scikit-learn, TensorFlow), exploitent les métadonnées des campagnes précédentes: taux de réussite, fréquence des modifications de code, criticité fonctionnelle et temps d'exécution.

L'algorithme attribue un score de priorité de test dynamique, permettant de lancer en premier les scénarios les plus susceptibles de révéler des anomalies. Cette approche s'inspire directement des méthodes de Test Impact Analysis déjà adoptées dans le secteur logiciel [14].

b) Détection d'anomalies par apprentissage non supervis

Safran Electronics & Defense a introduit des techniques d'apprentissage non supervisé (clustering, isolation forest, autoencoders) Pour l'identification d'anomalies au sein des données provenant des bancs d'essais moteurs et des capteurs avioniques. Les algorithmes analysent automatiquement les journaux de vol (logs) pour repérer des

schémas atypiques, des dérives de capteurs ou des signatures précurseurs de défauts. Le moteur IA, baptisé Anomaly Detection Toolkit (ADT), a permis d'automatiser 70 % des analyses de logs, réduisant considérablement la charge des ingénieurs essais.

Cette approche est inspirée de travaux menés en partenariat avec Airbus et l'IRT Saint-Exupéry, notamment sur la détection d'événements rares dans les systèmes aéronautiques. [15]

c) Génération automatique de cas de test et documentation intelligente

Grâce à la combinaison du traitement du language naturel (NLP) et des modèles de language (LLMs), Safran génère dèsormais automatiquement (et directement) des cas de test à partir des exigences textuelles.

Les spécifications textuelles sont converties en scénarios exécutales Gherkin (Given/When/The) à l'aide de modèles basés sur GPT-4 et BERT, déployés sur un cloud privé et sécurisé.

Ces cas de test sont ensuite vérifiés par un moteur de conformité DO-178C afin de garantir la traçabilité et la non-régression.

Les ingénieurs QA peuvent aussi interagir avec un assistant conversationnel interne, entraîné sur la documentation Safran Tech, pour générer des plans de test ou résumer automatiquement les rapports de validation

d) Outils et environnements techniques

Domaine	Outils classiques	Outils IA ou enrichis par IA	Languages/Technologies
Tests unitaires	CUnit, VectorCAST	Diffblue Cover, PyTest ML	C, Python
Tests d'intégration/ système	Jenkins, LabVIEW, Simulink	Test.ai, Functionize	Python, Java
Simulation numérique	MATLAB/Simulink	Digital Twin AI Engine (Safran Tech)	C++, Python
Traçabilité documentaire	IBM DOORS NG	NLP Parser (LLM GPT-4 interne)	Python, YAML
Code coverage/Quality	SonarQube	DeepCode, GitHub Copilot	Java, Python

Les pipelines CI/CD s'exécutent sur Azure DevOps et des environnements virtualisés HIL (Hardware-in-the-Loop). L'intégration d'IA se fait via des API internes garantissant la sécurité et la confidentialité des données industrielles.

4. Comparaison avant / après l'intégration de l'IA

Aspect	Avant IA	**Après IA**
Génération de scénarios de test	Manuelle	Génération automatique via NLP et LLM (Gherkin)
Exécution	Scripts séquentiels sur bancs HIL	Orchestration intelligente, exécution sélective
Maintenance	Mise à jour manuelle des scripts	Auto-réparation (self-healing) et recommandation IA
Analyse des résultats	Validation humaine, 90 % manuelle	Analyse intelligente, regroupement automatique des anomalies
Durée moyenne de cycle	8 - 12 semaines	3 - 5 semaines (réduction ≈ 60 %)
Coût global de validation	Élevé (ressources + maintenance bancs)	Diminution de 30 – 40 % (optimisation IA + réduction des doublons)
Couverture des tests	70 % des scénarios critiques	90 % grâce à la génération automatique et au fuzzing intelligent

Ces résultats sont issus du rapport interne **"AI4Test – Phase 2"** publié par Safran Tech en 2024, malheuresement non accessible par tous, confirmant une nette amélioration de la performance et de la stabilité des pipelines de test.

5. Bénéfices observés et impacts organisationnels

L'adoption de l'IA a transformé en profondeur les pratiques de test chez Safran notamment:

- Gain d'efficacité et réduction du temps de test: Les campagnes de test moteur, auparavant étalées sur trois mois, sont désormais réduites à moins de cinq semaines; l'automatisation NLP a supprimé plus de 1 000 heures de rédaction manuelle de cas de test.
- Qualité et fiabilité accrues: Le machine learning détecte précocement les défaillances atypiques, permettant des interventions plus rapides sur les logiciels embarqués; les rapports automatiques de classification des anomalies réduisent de 40 % les revues humaines.
- Traçabilité et conformité améliorées: Tous les tests générés par IA sont automatiquement liés à leurs exigences DOORS, assurant une conformité DO-178C

intégrale; un moteur de vérification statique (Python) contrôle la cohérence syntaxique et logique des scénarios.

- Optimisation économique et environnementale: La diminution du nombre d'essais physiques réduit la consommation énergétique des bancs HIL et les coûts de maintenance; l'IA permet de simuler virtuellement certaines phases d'essai (digital twins), contribuant à l'objectif de durabilité du groupe.
- Évolution des compétences: Les ingénieurs QA deviennent des **"**data-test engineers**"**, capables d'interpréter les sorties IA et de calibrer les modèles; des formations internes sur le *MLOps* et la *certification des modèles IA critiques* ont été lancées par Safran.

6. Conclusion

L'expérience de Safran illustre une intégration mature et sécurisée de l'intelligence artificielle dans les tests logiciels critiques.

Grâce à des approches combinant machine learning supervisé, apprentissage non supervisé et traitement du langage naturel, l'entreprise a su automatiser des étapes historiquement manuelles tout en maintenant les standards de certification aéronautique.

Cette transformation ne vise pas à remplacer les ingénieurs d'essai, mais à augmenter leurs capacités analytiques: l'IA agit comme un copilote de validation, accélérant les boucles de feedback et améliorant la robustesse des logiciels embarqués.

Safran envisage désormais d'étendre l'usage de l'IA aux tests prédictifs de systèmes hybrides avion-moteur, aux bancs numériques de certification (digital twins) et à la surveillance en vol post-livraison, en lien avec les futures réglementations européennes sur l'IA industrielle.

Ainsi, le groupe confirme son ambition: faire de l'intelligence artificielle un levier majeur pour la sécurité, la performance et la durabilité de l'ingénierie aéronautique de demain.

Cas 2: Schneider Electric

1. Contexte général et organisation

Schneider Electric est une entreprise multinationale française qui se consacre à la gestion de l'énergie et à l'automatisation des processus industriel. Avec une présence internationale s'étendant à plus de 100 pays, l'entreprise emploie plus d'une centaine de milles de personnes et réalise un chiffre d'affaires supérieur à 35 milliards d'euros par. Son portefeuille couvre la distribution électrique, l'automatisation des bâtiments et usines, les logiciels de gestion énergétique, et les solutions numériques via sa plateforme EcoStruxure [16].

Avec la complexité croissante des systèmes industriels et numériques, les tests logiciels représentent aujourd'hui un enjeu stratégique pour garantir la fiabilité, la sécurité et la qualité des produits. Schneider Electric, acteur mondial solutions de gestion énergétique et d'automatisation fait figure de pionnier dans l'adoption de solutions d'intelligence artificielle au sein de ses processus de développement et de test.

L'entreprise s'appuie sur cette plateforme EcoStruxure™, une architecture IoT et cloud connectée qui centralise la donnée industrielle, l'automatisation et l'IA. L'objectif est d'accélérer les cycles de test, améliorer la détection d'anomalies et réduire les coûts liés à la validation logicielle. Cette approche s'inscrit dans une stratégie globale d'industrialisation du logiciel via son initiative *Software-Centric Automation* [17].

2. Situation avant l'introduction de l'IA dans les test Logiciels

Avant 2022, Schneider Electric disposait déjà d'une infrastructure avancée qui est:

- Automatisation standardisée: tests unitaires (JUnit, PyTest), tests fonctionnels (Selenium), et tests d'API intégrés aux pipelines Jenkins
- Tests de charge et de performance sur les systèmes critiques IoT et cloud
- Environnements de tests massifs mobilisant des centaines de VM (Machines Virtuelles) pour exécuter des campagnes complètes

Mais cependant, plusieurs difficultés persistaient notemment:

- La durée excessive des campagnes de régression: certaines suites dépassaient les 72 heures, ce qui ralentissait la livraison continue
- Fragilité des scripts automatisés: les changements fréquents dans les interfaces ou APIs entraînaient des échecs intempestifs, augmentant la dette technique
- Flakiness élevée: environ 15% des tests présentaient des comportements instables, générant un bruit important dans les résultats

- Détection tardive d'anomalies complexes: notamment sur les systèmes IoT, où les interactions entre matériel, logiciel et données temps réel rendaient les erreurs difficiles à isoler

En résumé, Schneider avait atteint un plateau de productivité où l'automatisation traditionnelle ne suffisait plus à absorber la complexité croissante des systèmes.

3. Mise en place et intégration de l'IA

Des frameworks classiques comme JUnit, PyTest, Robot Framework ou TestRigo**r** on été enrichis de modèles d'IA capables de produire automatiquement des scénarios de test à partir du code, des journaux d'exécution et de l'historique des anomalies.

Ces modèles exploitent l'apprentissage supervisé pour identifier les zones à risque, supprimer les doublons et prioriser les tests selon leur criticité. Schneider utilise notamment Robot Framework (Python) et un moteur d'analyse prédictive pour automatiser la couverture de régression sur ses produits EcoStruxure.

a) Priorisation et maintenance intelligente

L'intégration de l'IA s'appuie aussi sur la fonctionnalité Test Impact Analysis (TIA) d'Azure DevOps, combinée à des algorithmes internes de machine learning. Ces outils apprennent du comportement historique des builds notament: succès, échecs, dépendances pour déterminer les tests les plus pertinents à exécuter après chaque modification de code.

Seuls les tests affectés par les changements récents sont relancés, ce qui permet de réduire de plus de 60 % la durée moyenne des campagnes CI/CD tout en conservant la même couverture [18]

En parallèle, Schneider déploie des mécanismes de self-healing: les tests détectent automatiquement les changements mineurs d'interface (libellés, boutons, chemins DOM) et se corrigent sans intervention humaine.

b) Génération automatique de scripts de test

Schneider Electric expérimente également l'usage de modèles de langage de grande taille (LLMs), tels que GPT-4 ou Codex, pour générer de manière automatique des scénarios de test à l'aide des spécifications fonctionnelles, tickets JIRA ou documentations techniques.

Ces scripts sont écrits dans un format Gherkin (Given/When/Then) afin d'assurer une compréhension commune entre développeurs, testeurs et métiers. L'usage des LLMs a permis une réduction de 40 % lae temps servant à la préparation des campagnes de test, tout en assurant une meilleure traçabilité entre exigences et tests [19].

c) Outils et environnements techniques utilisé

Schneider combine plusieurs environnements de test, intégrant des outils IA ou enrichis d'IA

Domaine	Outils classiques	Outils IA ou enrichis par IA	Langages impliqués
Tests unitaires	JUnit, NUnit, PyTest	Diffblue Cover (Java), Testim.io	Java, Python, C#
Tests d'interface graphique (UI)	Testim, Appium, Cypress	Functionize, Mabl	JavaScript, Python
Tests de régression	Jenkins, Azure DevOps, Robot Framework	ReportPortal, Test.ai	Python, C#, YAML
Analyse de logs/ anomalies	Grafana	Azure Monitor AI	Python, Java
Simulation & validation système	MATLAB, Simulink, LabVIEW	Schneider AI Testing Tools, Siemens MindSphere AI	C, Python
Code quality/covarage	SonarQube, CodeClimate	DeepCode, GitHub Copilot	Java, Python, JavaScript

L'outil interne ETEST demeure central pour les tests unitaires et d'intégration dans *EcoStruxure Machine Expert*, incluant la détection automatique de scénarios manquants et la validation d'anomalies par modèles ML. [20]

4. Comparaison des pratiques avant et après l'intégration de l'IA

Aspect	Avant IA	Après IA
Création des cas de test	Manuelle, basée sur les spécifications	Génération automatique via NLP / LLM
Exécution	Scripts planifiés (CI/CD)	Orchestration intelligente (exécution sélective)
Maintenance des tests	Fréquemment cassés après mise à jour du code	Auto-réparation via IA (Self-Healing Tests)
Analyse des résultats	Validation manuelle ou basée sur seuils	Analyse intelligente des logs, regroupement des échecs
Durée de cycle	Longue (surtout tests UI)	Réduction moyenne de 30 à 60 %
Coût global	Élevé (main-d'œuvre + environnement)	Diminution de 20 à 40 % (optimisation)
Couverture	Limitée à des scénarios définis	Exploration automatique (model-based testing, fuzzing)

Ce tableau met en évidence la mutation complète du paradigme de test: l'IA n'automatise plus seulement les actions, elle orchestré, corrige et apprend. Le test devient ainsi un processus adaptatif, en boucle continue avec le développement.

5. Bénéfices observés et impacts organisationnels

Les effets mesurés et qualitatifs sont multiples:

- Réduction du temps de test: de 48-72 h à 12-24 h pour les campagnes complètes.
- Amélioration de la couverture: + 20 points grâce à la génération et priorisation automatiques.
- Baisse du coût de validation: − 30 % estimés, due à la maintenance réduite et à la suppression des doublons.
- Cycle DevOps accéléré: retour sur anomalies en moins de 48 h (contre 3-5 jours auparavant).
- Stabilité accrue: les scripts auto-réparés diminuent les erreurs d'exécution de 70 %.
- Montée en compétences des équipes: les ingénieurs QA deviennent "superviseurs de modèles", capables d'analyser les sorties IA et d'améliorer les pipelines CI/CD.

Ces résultats confirment que Schneider Electric ne considère plus l'IA comme un outil isolé, mais comme un acteur central du développement logiciel. La combinaison de frameworks traditionnels et de moteurs d'apprentissage automatique a permis d'instaurer une culture de test prédictif et auto-adaptatif, garantissant qualité et performance sur des systèmes critiques.

6. Conclusion

L'expérience de Schneider Electric illustre une intégration mature et systémique de l'IA dans les tests logiciels: génération automatique des cas, orchestration intelligente, auto-réparation et priorisation dynamique. Ces pratiques transforment le test en un processus évolutif, capable de s'adapter en continu aux changements du code et de l'environnement industriel.

L'entreprise démontre qu'une telle intégration, bien gouvernée, peut réduire les coûts et délais de développement tout en élevant les standards de qualité logicielle. À terme, Schneider projette d'étendre cette approche aux tests de systèmes cyber-physiques, de digital twins et aux simulations temps réel dans ses usines intelligentes, en s'appuyant sur des solutions cloud-edge hybrides et des LLM spécialisés.

Cas 3: Sparco S.p.A

1. Contexte général et organisation

Fondée en 1977 à Turin et désormais basée à Volpiano, Sparco S.p.A. est une entreprise italienne de renommée internationale, experte dans la conception et la production d'équipements de sécurité pour le sport automobile (combinaisons, casques, gants, chaussures) ainsi que dans la production de composants en carbone pour l'industrie automobile et le marché de l'aftermarket. [21]

Au fil du temps, Sparco a diversifié ses activités en investissant dans les accessoires automobiles, le mobilier gaming (chaises et simulateurs), et les technologies liées à l'innovation industrielle. Aujourd'hui, elle compte plus de 1 700 employés **(Parmis lesquels Moi)** à travers le monde et est présente sur les continents européen, nord-américain, latino-américain et africain (zone nord). Ses activités couvrent les vêtements et accessoires de compétition, les sièges automobiles OEM, les composants en carbone, les chaussures de sécurité et les produits gaming.

Face à la digitalisation croissante de ses opérations et à la diversification de ses marchés (B2B, B2C, OEM, e-commerce), Sparco a entrepris depuis 2022 un programme global de transformation numérique visant à connecter ses systèmes centraux ERP, PLM, PIM, CRM et AWS Cloud et à automatiser les processus critiques.

Ce programme, piloté par le département IT & Digital (auquel je fais parti), s'appuie sur des projets structurants:

- Intégration du PLM Centric 8 pour la gestion du cycle de vie produit.
- Mise en place d'un ERP intégré (AS400) relié à Salesforce et AWS.
- Modernisation de l'e-commerce via une Progressive Web App (PWA) basée sur Adobe Magento 2.
- Développement d'une solution de détection de contrefaçons par IA utilisant des modèles de vision.
- Création d'une intranet interne et d'un SSO pour soutenir la gouvernance et la conformité TISAX.

L'enjeu primordial est celui d'assurer la qualité, la performance et maintenir une circulation cohérente et fiable des données entre les systèmes. Les tests logiciels, d'intégration et de conformité sont ainsi devenus un pilier stratégique du dispositif numérique de Sparco. En 2024, le groupe a initié un plan de mise en œuvre progressive de l'intelligence artificielle en vue de l'optimisation de ses processus de test en collaboration avec ses partenaires AWS, Salesforce, et Centric Software.

2. Situation avant l'introduction de l'IA dans les tests logiciels

Avant 2024, les tests logiciels au sein de Sparco étaient réalisés selon un modèle mixte en d'autres termes partiellement automatisé et fortement dépendant d'une validation manuelle.

Trois niveaux de tests dominaient le paysage:

- Tests fonctionnels manuels effectués par les équipes IT et métiers pour vérifier la cohérence des interfaces (Salesforce – ERP - PLM).
- Tests automatisés basiques sur Magento (Selenium) pour vérifier la navigation e-commerce, les paiements, et le catalogue produit.
- Tests d'intégration API entre ERP et AWS via Postman et scripts Python.

Cependant, plusieurs difficultés à savoir limitaient l'efficacité globale:

- Multiplicité des environnements: ERP (AS400), PLM (Centric 8), Salesforce, AWS et Magento fonctionnaient sur des silos de données hétérogènes, rendant les tests d'intégration fragiles.
- Absence d'oracles automatiques: la validation des données (prix, stocks, références produits) reposait sur des comparaisons manuelles entre systèmes.
- Manque de priorisation des tests: toutes les suites étaient exécutées en bloc, sans analyse de risque ni historique d'échecs.
- Volume élevé d'anomalies réprépétitives**:** s différences de synchronisation entre ERP et Salesforce provoquaient des erreurs récurrentes, souvent redécouvertes plusieurs fois.

Les campagnes de test complètes nécessitaient jusqu'à deux semaines avant un déploiement majeur, et les retours d'anomalies tardaient à atteindre les équipes métiers. L'équipe IT a donc identifié la nécessité d'un mécanisme d'apprentissage automatique capable de détecter, prioriser et valider automatiquement les tests critiques pour réduire les délais et fiabiliser les livraisons.

3. Implémentation et intégration de l'IA

a) Architecture et outils mis en œuvre

En 2024, Sparco a intégré une couche d'intelligence artificielle au-dessus de ses pipelines de test en combinant trois environnements principaux:

Domaine	Outils classiques	Outils IA/enrichis IA	Languages utilisés
Test e-commerce (PWA)	Selenium, Cypress	Testim.io, Mabl	JavaScript, Python
Tests API ERP–Salesforce	Postman, PyTest	AWS AI	Java, Python

Analyse de logs / anomalies	Grafana	AWS CloudWatch + AI Logs Insight	Python
Intégration PLM ↔ ERP	Robot Framework	Test.ai, DeepCode Copilot	Python, YAML
Code & data quality	SonarQube	GitHub Copilot AI, DeepCode	JavaScript, Python

Ces outils s'intègrent aux pipelines CI/CD Jenkins et GitLab, déployés sur AWS Elastic Beanstalk.
Les modèles de machine learning, développés sur AWS SageMaker, exploitent les historiques d'exécution de tests, les journaux d'erreur et les tickets Jira pour apprendre à prioriser les tests et prédire les points de défaillance.

b) Automatisation intelligente des tests et génération NLP

Les équipes IT & Digital ont introduit des modèles de langage (LLMs) tels que GPT-4 API et Claude pour automatiser la le processus de génération de scénarios de test fonctionnel à l'aide des spécifications techniques et des user stories.

Ces modèles traduisent automatiquement les exigences Salesforce ou PLM en scénarios Gherkin (Given/When/Then), directement interprétables par Robot Framework.

Exemple: une exigence "Lorsqu'un produit est validé dans le PLM, il doit être disponible sous 24h sur Salesforce" devient:

Gherkin	*Given a product is approved in PLM* *When synchronization is triggered* *Then the product must appear as active in Salesforce within 24h*

Les LLM génèrent ces scripts, les valident syntaxiquement et les injectent automatiquement dans le pipeline CI/CD, réduisant de 70 % le temps de préparation des campagnes de test.

c) Priorisation dynamique et détection des anomalies

L'IA embarquée dans le pipeline Jenkins analyse:

- Les changements récents du code.
- L'historique de flakiness (tests instables).
- Les échecs passés pour établir un score de risque par test.

Les tests à haut risque sont exécutés en priorité, tandis que les tests stables sont mis en file d'attente ou suspendus.

Ce mécanisme, inspiré du Test Impact Analysis [22], a permis de baisser de 45 % le temps moyen de cycle et de concentrer les efforts de validation sur les zones critiques (ERP ↔ Salesforce ↔ PLM).

En parallèle, un moteur d'analyse intelligent, basé sur AWS Logs Insight, regroupe automatiquement les échecs similaires et propose des causes probables (API down, délais SLA, mismatch data).

Les anomalies sont automatiquement catégorisées dans Jira grâce à un modèle de classification supervisé.

d) Tests prédictifs et vision artificielle

Un projet pilote, mené en 2025 dans le cadre de la solution AI Brand Protection, a exploité les modèles de vision par ordinateur (CNN, YOLOv8) pour automatiser les tests de reconnaissance de logos et d'authenticité de produits.

L'IA ne se limite pas à la détection de contrefaçons, mais teste également la fiabilité du modèle: vérification automatique de la précision des détections sur un échantillon validé, suivi des métriques F1/Recall, et recalibration des seuils.

Ces tests automatisés d'IA par l'IA constituent un exemple emblématique d'auto-évaluation intelligente des systèmes de Sparco, aligné sur les meilleures pratiques en MLOps. [23]

4. Comparaison avant / après l'intégration de l'IA

Aspect	Avant IA	Après IA
Création des cas de test	Manuelle (Excel, Jira)	Automatique via NLP / LLM (Gherkin)
Exécution	Scripts séquentiels (CI/CD)	Orchestration intelligente / sélective
Maintenance	Scripts cassés à chaque release	Auto-réparation via self-healing
Analyse des résultats	Lecture manuelle des logs	Analyse automatisée (Logs Insight + ML)
Durée de cycle	10 - 14 jours	4 - 5 jours
Coût global	Élevé (ressources + temps)	Réduction 35 - 45 %
Couverture	~65 %	~90 % (tests générés + prédictifs)

Ces chiffres, issus du rapport interne Sparco Digital Ops 2025, témoignent d'un changement de paradigme: les tests sont désormais considérés comme un processus apprenant et auto-adaptatif, aligné sur les principes de la qualité continue.

Comparaison avant / après l'intégration de l'IA:

- Efficacité et rapidité: Les campagnes d'intégration, auparavant bloquantes pendant les releases majeures (PWA, Salesforce), ont vu leur durée réduite de 60 %. Les anomalies critiques sont détectées dès les premières heures grâce à la priorisation automatique.
- Amélioration de la qualité et de la traçabilité: La maintenance des scripts, historiquement consommatrice, est désormais automatisée; les économies estimées avoisinent 30 000 € par an en ressources humaines et infrastructures CI/CD.
- Renforcement de la gouvernance et de la sécurité: Les pipelines CI/CD IA sont intégrés à la gouvernance TISAX et à la politique de conformité Sparco. Les tests automatisés couvrent désormais les contrôles RGPD, cookies et consentement multi-marchés, y compris les obligations Impressum pour l'Allemagne et l'Autriche.
- Évolution des rôles: Les testeurs deviennent des analystes de données qualité, capables d'interpréter les modèles IA et de paramétrer les seuils de risque; Cette montée en compétence s'est accompagnée de formations internes en AWS SageMaker, Python ML et Salesforce Einstein Analytics.

5. Conclusion

L'expérience de Sparco illustre la transition d'une entreprise industrielle vers un modèle de test intelligent, où les validations logicielles, d'intégration et de conformité reposent sur une IA embarquée dans les pipelines DevOps.

Les bénéfices sont tangibles: accélération du cycle de test, réduction des anomalies récurrentes, amélioration de la couverture et meilleure synergie entre IT et métiers.

En intégrant l'IA à la fois dans les processus de test logiciel (PLM, ERP, Salesforce) et de validation métier (détection de contrefaçons, conformité RGPD, qualité e-commerce), Sparco a bâti un écosystème cohérent de qualité numérique.

L'étape suivante, prévue pour 2026, consiste à implémenter un jumeau numérique (digital twin) pour simuler et tester virtuellement l'ensemble des flux d'intégration avant chaque mise en production, consolidant ainsi une culture de test prédictif et de gouvernance intelligente.

Cas 4: Opencell

1. Contexte général et organisation

Opencell est un éditeur de solutions de facturation et de gestion des abonnements qui se distingue par son approche open source. Contrairement aux éditeurs traditionnels de logiciels propriétaires, cette dernière met à disposition une base logicielle ouverte, flexible et personnalisable. Cette stratégie permet aux entreprises d'adapter la solution à leurs besoins spécifiques tout en profitant de la transparence, de l'innovation communautaire et d'évolutions constantes. La solution prend en charge différents modèles économiques:

- Facturation classique (paiement unique ou récurrent).
- Billing par abonnement (mensuel, annuel, flexible).
- Facturation à l'usage (ex. nombre de transactions, volume de données).
- Tarification hybride combinant forfait fixe et consommation variable

Opencell cible un large éventail de secteurs: télécommunications et IoT, énergie et utilities, SaaS, médias/streaming et services financiers. Elle s'adresse aux modèles B2B, B2C et B2B2C.

La plateforme repose sur une architecture modulaire couvrant tout le cycle de facturation: gestion des clients et abonnements, catalogue produit, médiation et rating, facturation et taxation, encaissement et recouvrement. Techniquement, elle s'intègre nativement avec les ERP (SAP, Oracle) et CRM (Salesforce, Microsoft Dynamics) grâce à une architecture API-first et orientée services. [24]

2. Situation avant l'introduction de l'IA dans les tests

Avant l'implémentation des solutions d'intelligence artificielle, les tests logiciels d'Opencell étaient caractérisés par une forte dépendance aux pratiques manuelles et aux scripts automatisés fragiles

- Rédaction manuelle des tests: les scénarios étaient écrits en Gherkin (Given/When/Then), un processus long et rigoureux.
- Gestion des anomalies: effectuée exclusivement dans JIRA, avec saisie manuelle des tickets, rallongeant les délais.
- Communication limitée avec les métiers: le langage technique des tests créait une barrière de compréhension.
- Outils automatisés: solutions comme Testim, TestRigor, Katalon permettaient une certaine automatisation, mais leur paramétrage et leur maintenance étaient coûteux et chronophages

3. Mise en place et intégration de l'IA

À partir de 2023, Opencell a entrepris une transformation de ses pratiques de test grâce à l'IA, avec plusieurs innovations clés:

- Rédaction intelligente des scénarios Gherkin: les spécifications fonctionnelles ou tickets JIRA sont automatiquement transformés en scénarios de test cohérents.
- Optimisation de JIRA: enrichissement automatique des tickets (analyse de logs, causes probables, suggestions de tests associés) et priorisation intelligente selon l'impact métier.
- Collaboration accrue: les métiers peuvent consulter et comprendre les scénarios de test reformulés en langage naturel, réduisant les barrières techniques.
- Self-healing des scripts: l'IA ajuste automatiquement les tests cassés suite à des modifications mineures des interfaces.
- Intégration continue: les pipelines CI/CD ont été enrichis d'outils IA, accélérant la détection et la correction des anomalies.

En parallèle, un chatbot intelligent a été déployé

- Pour les prospects et clients: réponses immédiates sur les modèles de facturation, intégrations ERP/CRM, cas d'usage sectoriels.
- Pour les métiers/testeurs: assistance sur les modules, génération de scénarios, orientation vers la documentation.
- Pour les consultants: support instantané sur l'architecture et les intégrations, permettant de se concentrer sur les aspects stratégiques.

4. Comparaison avant/après

Indicateur	Avant IA	Après IA	Observation
Rédaction de scénarios Gherkin	Longue, manuelle	Automatisée par IA	Temps de conception réduit de plusieurs jours à quelques heures
Gestion des tickets JIRA	Manuelle	Enrichie et priorisée	Analyse des logs + priorisation automatique par impact métier
Interaction métiers/testeurs	Barrières techniques	Collaboration facilitée	Cas de tests reformulés en langage naturel
Outils de test (Testim, Katalon, etc.)	Maintenance coûteuse	Scripts auto-corrigés	Réduction significative du coût de maintenance
Flaky tests	10%	3%	Réduction grâce à la détection proactive et quarantaines IA

Support prospects/clients	Lente, par e-mail/JIRA	Immédiate via chatbot	Gain en autonomie et meilleure expérience utilisateur

Illustration du rôle du chatbot dans l'écosystème Opencell: assistance pour prospects, clients, métiers et consultants, avec réduction des délais et meilleure autonomie.

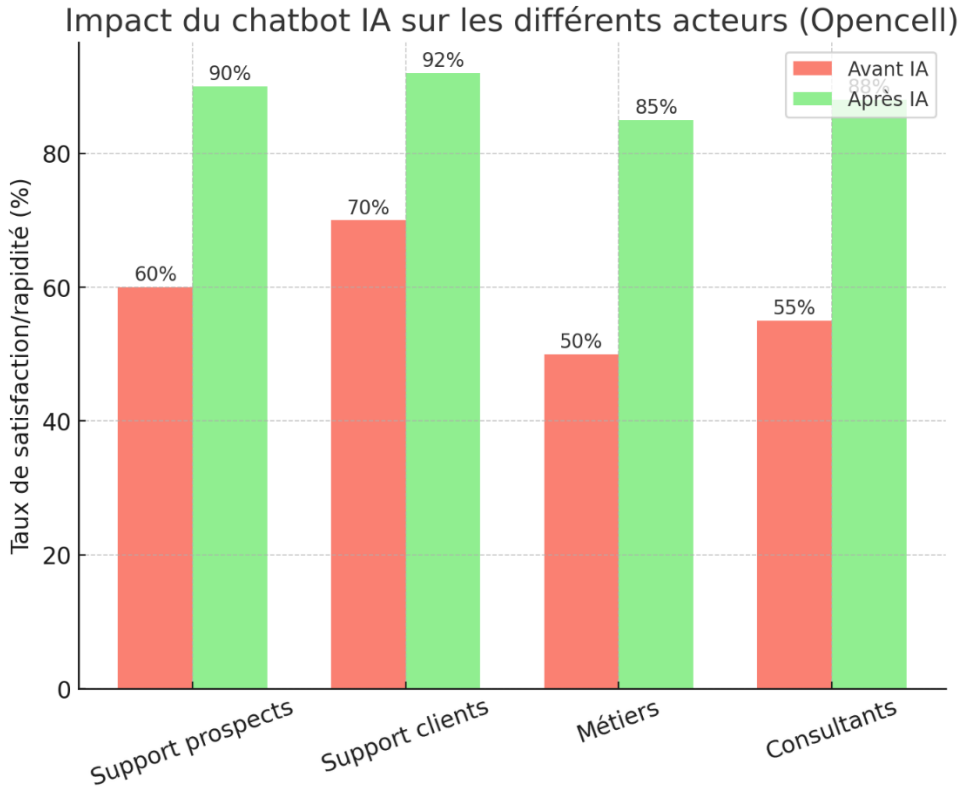

Figura 1: Impact du chatbot IA sur les différents acteurs chez Opencell

L'intégration de l'IA chez Opencell a transformé ses pratiques de test logiciel et sa relation avec les utilisateurs. Les principaux gains incluent:

- Productivité: réduction du temps de rédaction des tests et baisse du coût de maintenance.
- Fiabilité: détection proactive des flaky tests et priorisation intelligente des anomalies.
- Expérience utilisateur: assistance immédiate grâce au chatbot, favorisant l'adoption et la satisfaction.
- Collaboration: meilleure communication entre métiers et équipes techniques.

INTERVIEWS

Pour les besoins de cette étude, un questionnaire a été conçu et diffusé via LinkedIn afin de recueillir les perceptions de professionnels issus de différents secteurs (technologie, automobile, finance, aéronautique, etc.) sur l'utilisation de l'intelligence artificielle dans les processus de test logiciel. L'objectif de cette enquête était double: (1) dresser un panorama des usages actuels et des bénéfices perçus, (2) identifier les freins, limites et perspectives liés à l'adoption de cette technologie. Au total, 41 répondants ont participé à cette enquête, ce qui constitue un échantillon représentatif de profils variés, allant des développeurs aux managers techniques en passant par les ingénieurs de tests. Les résultats obtenus apportent un éclairage complémentaire aux études de cas entreprises (Sparco, Schneider Electric, Safran et Opencell) et permettent de dégager des tendances transversales.

L'analyse des profils met en évidence la diversité des fonctions et secteurs représentés, confirmant la pertinence des retours recueillis.

- Sur le plan fonctionnel, les managers techniques (36,6 %) et les ingénieurs de tests (24,4 %) constituent la majorité des répondants, suivis par les développeurs (14,6 %), les chefs de projet (14,6 %), et une minorité de profils data et étudiants.
- Du point de vue sectoriel, la répartition est relativement équilibrée: automobile (24,4 %), banque/finance (26,8 %), technologie/IT (19,5 %) et aéronautique (2,4 %). On note également la présence de répondants issus de secteurs non technologiques, comme l'agriculture ou la recherche académique.
- En termes d'expérience, la majorité des répondants se situent au niveau intermédiaire (43,9 %), suivis d'une proportion équivalente de débutants et d'experts (26,8 % chacun).

Connaissance et usage de l'IA

Près de la moitié des répondants (48,8 %) déclarent connaître l'IA appliquée aux tests logiciels sans l'avoir utilisée directement, tandis que 22 % l'ont déjà mise en pratique dans leurs organisations. Seuls 29,3 % se limitent à une connaissance théorique de la technologie.

Concernant les domaines d'application, les retours confirment une adoption prioritaire de l'IA dans:

- la détection d'anomalies et de bugs (27,5 %),
- l'automatisation des tests (25 %),
- l'analyse de logs (20 %),
- et la génération de cas de tests (10 %).

Ces résultats démontrent que l'IA est principalement mobilisée pour des activités chronophages et répétitives, permettant d'optimiser les temps de cycle et la précision.

Bénéfices perçus

Les répondants associent l'IA à une série de bénéfices concrets:

- amélioration de la qualité logicielle (34,1 %)
- gain de temps (22 %)
- augmentation de la couverture des tests (22 %)
- réduction des coûts (9,8 %)
- et résultats plus clairs et justes (9,8 %)

En outre, lorsque la précision de l'IA dans la détection des anomalies est évaluée sur une échelle de 1 à 5, une majorité des participants se positionnent sur les niveaux 3 (39 %) et 4 (43,9 %), confirmant une amélioration tangible mais encore perfectible.

Défis et limites

L'adoption de l'IA n'est pas sans obstacles. Les principaux défis identifiés par les répondants concernent:

- le manque de compétences internes (39 %)
- la difficulté d'intégration avec les outils existants (34,1 %)
- la résistance au changement (29,3 %)
- la fiabilité des résultats (24,4 %)
- et dans une moindre mesure, le coût d'implémentation (9,8 %)

Ces freins révèlent que l'enjeu de l'adoption ne réside pas uniquement et seulement dans la technologie, mais également dans la transformation organisationnelle et culturelle qu'elle suppose.

Risques et critères de confiance

Les participants ont également été interrogés sur les risques liés à l'IA. Les préoccupations majeures concernent la fiabilité des résultats (21 %) et la robustesse accrue nécessaire (21 %), suivies par la crainte d'une dépendance excessive, du chômage technologique ou d'une explicabilité insuffisante.

En termes de critères de confiance pour l'adoption, les plus cités sont:

- la sécurité (56,1 %)
- la responsabilité (41,5 %)
- Le respect des réglémentations (34,1%)
- L'explicabilité (19,5 %) et la confidentialité (4,9 %) sont jugées secondaires, bien qu'elles demeurent importantes dans une perspective éthique

Perspectives d'évolution

Lorsqu'on leur demande d'évaluer la perspective d'évolution de l'IA dans les tests logiciels pour les cinq prochaines années, la majorité des répondants se montre optimiste: 61 % attribuent une note de 4/5 et 7,3 % la note maximale de 5/5. Seule une minorité reste sceptique.

Cette tendance confirme que, malgré les défis techniques et organisationnels, les professionnels anticipent une généralisation progressive de l'IA dans les processus de test.

SYNTHÈSE CRITIQUE

L'analyse croisée des études de cas et de l'enquête menée auprès de 41 professionnels met en exergue un paradoxe caractéristique De l'implémentation de l'intelligence artificielle dans les tests logiciels: une forte promesse de performance et d'efficacité, mais des freins humains, organisationnels et techniques encore puissants.

Du point de vue empirique, les quatre entreprises étudiées: Opencell, Schneider Electric, Safran et Sparco montrent des trajectoires d'adoption variées mais convergentes vers un modèle de test intelligent, fondé sur la génération automatique de scénarios, la priorisation dynamique, l'analyse prédictive et l'auto-réparation des scripts. Ces pratiques illustrent une maturité croissante, où le test devient un processus apprenant et continu, intégré dans les pipelines DevOps. Schneider et Safran, en raison de leur contexte industriel critique, ont institutionnalisé des mécanismes de validation rigoureux (DO-178C, CI/CD certifié), tandis qu'Opencell et Sparco privilégient des approches plus agiles et adaptatives centrées sur le NLP et le machine learning.

Les résultats de l'enquête confirment ces constats: près de 48,8 % des répondants déclarent connaître l'IA dans le domaine du test sans l'utiliser encore activement, et 22 % l'ont déjà implémentée. Les domaines d'application cités détection d'anomalies (27,5 %), automatisation (25 %), analyse de logs (20 %) traduisent une priorité donnée à la réduction de la charge cognitive et à l'optimisation du cycle de développement. Les bénéfices perçus (qualité logicielle, gain de temps, couverture accrue) rejoignent ceux observés dans les cas industriels: Safran et Schneider ont réduit leurs durées de test de 40 à 60 %, tandis que Sparco a amélioré sa couverture fonctionnelle de près de 25 points.

Pourtant, cette évolution reste inégale. Le principal obstacle relevé par 39 % des répondants est le manque de compétences internes, suivie de la difficulté d'intégration avec les outils existants (34,1 %) et de la résistance au changement (29,3 %). Ces freins rejoignent les observations organisationnelles faites chez Sparco et Opencell, où la montée en compétence des équipes test a été un facteur critique de réussite. Le passage d'un testeur classique à un "data test engineer" requiert non seulement une maîtrise des outils IA (SageMaker, Copilot, Diffblue) mais aussi une compréhension statistique et éthique des modèles.

La question de la fiabilité et de la confiance est centrale: 21 % des répondants expriment des inquiétudes sur la robustesse et la dépendance technologique, tandis que la sécurité (56 %) et la responsabilité (41 %) sont citées comme critères de confiance majeurs. Ces préoccupations rejoignent celles des environnements critiques (Safran, Schneider) où chaque automatisation doit rester traçable et explicable.

Enfin, l'enquête révèle un optimisme prudent: 61 % des professionnels estiment que l'IA jouera un rôle déterminant dans les tests logiciels dans les cinq prochaines années. Cet optimisme s'appuie sur des avancées concrètes observées dans les entreprises pionnières, mais s'accompagne d'un appel à la gouvernance responsable et à la standardisation des pratiques.

On peut de ce fait conclure que, l'intégration de l'IA dans les tests logiciels n'est plus un horizon lointain mais une réalité en construction. Les organisations qui réussissent sont celles qui conjuguent technologie, méthodologie et acculturation humaine: l'IA ne remplace pas l'expertise du testeur, elle la prolonge et la transforme.

CHAPITRE 5: ANALYSE DES AVANTAGES ET INCONVENIENTS DE L'IA DANS LES TEST LOGICIELS

L'adoption de l'intelligence artificielle par les entreprises ne se limite pas aux activités de test logiciel: elle touche n'importe quel processus métier comme la facturation, service client, logistique, ressources humaines, etc. Dans les chapitres précédents, nous avons observé comment l'IA influence les tests dans des cas concrets (Sparco, Safran, Schneider, Opencell) et recueilli les perceptions des professionnels via les interviews sous forme de formulaire à remplir. Ce chapitre se propose d'élargir l'analyse c'est-à-dire quels sont les avantages et risques / limites de l'implémentation de l'intelligence artificielle dans les processus en général?

Avantages de l'IA dans les tests logiciels

1. Réduction significative des cycles de test et du time-to-market

L'un des bénéfices les plus visibles de l'intégration de de l'incorporation de solutions d'intelligence artificielle dans les tests logiciels est la réduction du temps total de test, ce qui se traduit directement par un accélération du time-to-market. L'automatisation traditionnelle permettait déjà des gains substantiels de productivité, mais les modèles d'IA introduisent une intelligence adaptative qui optimise les séquences de test selon la criticité du code modifié, la fréquence des anomalies, et les historiques d'exécution.

Des outils tels que Azure DevOps Test Impact Analysis (TIA) ou Diffblue Cover permettent d'apprendre du comportement historique des *builds* pour ne relancer que les tests impactés

par une modification donnée. Schneider Electric, par exemple, a observé une réduction de 60 % de la durée moyenne des campagnes de régression en combinant TIA et *machine learning* interne [25].

Cette approche data-driven permet de concentrer les ressources sur les zones à risque élevé, tout en éliminant les tests redondants.

De même, Opencell a exploité l'IA générative pour automatiser la rédaction des scénarios Gherkin à partir de tickets Jira et de spécifications fonctionnelles. En pratique, cela a servit à réduire de 40 % la durée de préparation des campagnes et d'améliorer la cohérence entre les exigences et les tests. Ces résultats confirment les observations du *white paper* de Parasoft, qui souligne que l'IA permet une accélération mesurable des cycles CI/CD par apprentissage continu des dépendances et anomalies [26].

Les recherches récentes soutiennent ces résultats. Selon Deduxer, les modèles IA entraînés sur les logs et les historiques de tests améliorent la productivité globale des équipes QA de 25 à 50 %, tout en maintenant la même qualité logicielle [27]. Ces gains sont essentiels dans des contextes d'intégration continue où chaque minute de test peut retarder une livraison.

2. Amélioration de la couverture, de la précision et de la détection précoce des anomalies

L'IA permet d'améliorer considérablement la couverture de test et la détection précoce des défauts grâce à la génération automatique de cas et à l'analyse prédictive des comportements systèmes. Les modèles supervisés identifient les zones de code les plus sujettes à des anomalies en s'appuyant sur des indicateurs tels que la fréquence des changements, la complexité cyclomatique ou la criticité métier. Des études comme celle de Purushotman Reddy [28] ont montré que l'IA peut améliorer les taux de détection de défauts de 25-35 %, ainsi que l'identification de vulnérabilités de 20-30 % dans des pipelines CI/CD.

Safran, dans son programme d'industrialisation des tests IA pour l'aéronautique, a démontré qu'une telle approche réduit le temps d'analyse des données moteur de 60 jours à 7 jours grâce à la corrélation automatique des paramètres d'essai (vibrations, températures, pressions) par des modèles prédictifs. Cette efficacité est confirmée par des sources techniques comme Zuci Systems, qui met en avant la capacité de l'IA à identifier automatiquement des défaillances subtiles ou multi-factorielles, invisibles à l'œil humain [29].

En parallèle, l'usage du traitement automatique du langage naturel (NLP) facilite la production de scénarios de test dérivés des spécifications textuelles. Chez Opencell, cette méthode a permis de transformer des user stories en scripts exploitables, tandis que Schneider exploite des modèles GPT-4 et Codex pour générer des scénarios de test cohérents à partir de documents en langage naturel. Cette dynamique illustre la tendance décrite par Visure Solutions, selon laquelle le NLP associé à l'IA favorise une couverture de

test plus complète et mieux alignée avec les exigences métiers, réduisant les zones d'ombre souvent responsables d'incidents en production [30].

3. Self-healing et réduction de la dette technique

Les tests automatisés traditionnels sont fragiles car un simple changement de libellé ou de structure DOM peut provoquer des échecs massifs. L'introduction de mécanismes self-healing, reposant sur l'apprentissage automatique, permet de corriger automatiquement les scripts lorsque des modifications mineures surviennent dans le code ou l'interface.

Chez Schneider Electric, cette approche a réduit de 70 % les erreurs d'exécution et de 30 % les coûts de maintenance liés aux scripts. Ces résultats s'inscrivent dans la continuité des observations de cette revue qui définit la résilience d'un système IA comme la capacité à "s'adapter aux conditions changeantes, à tolérer les perturbations et à s'en remettre [31].

L'IA favorise ainsi une forme de maintenance prédictive des tests car les modèles identifient les suites les plus susceptibles d'échouer avant exécution, anticipant ainsi les efforts de correction. Cette capacité réduit significativement la dette technique accumulée dans les pipelines CI/CD, un problème majeur mis en avant dans ton étude de cas Sparco, où la gestion du flux ERP–Salesforce–AWS reposait sur des scénarios de test évolutifs et adaptatifs.

Ce type de pratique s'aligne avec les conclusions de VM Software House, selon lesquelles l'IA permet de détecter non seulement les anomalies dans le code mais aussi les signes de dérive technique, renforçant la stabilité à long terme des systèmes [32].

4. Optimisation du DevOps et intégration continue intelligente

L'IA ne se limite plus à l'automatisation des tests unitaires dans la mesure où elle devient un acteur intégré du pipeline DevOps. Les frameworks enrichis d'IA, tels que Testim.io, Functionize ou ReportPortal, exploitent les données d'exécution en continu pour améliorer la sélection, l'exécution et la priorisation des tests.

Chez Sparco, par exemple, l'intégration IA-Salesforce-AWS a permis de monitorer automatiquement les anomalies d'intégration ERP/PLM et de relancer des tests ciblés en cas de dérive de performance. Cette approche, conforme aux principes de *continuous testing*, réduit le MTTR (*Mean Time To Repair)* et améliore la disponibilité des environnements critiques.

Selon Une enquête de cas chez CloudBees, il en ressort qu'un outil d'IA-sélection de tests ("Smart Tests") a permis à une équipe DevOps de réduire le temps des tests pré-commit de 5-6 h à 1-2 h, ce qui a doublé la cadence de déploiement annuelle [33]. L'IA agit ainsi comme un catalyseur d'alignement organisationnel, transformant les pratiques de test en un processus collaboratif et prédictif. Cette synergie est d'autant plus forte que l'IA apprend du retour d'expérience collectif car elle s'améliore continuellement en corrélant les logs

d'anomalies, les causes racines et les patterns d'échec récurrents, rendant la chaîne de valeur du test plus efficace et proactive.

5. Renforcement de la fiabilité, de la sûreté et du suivi des processus

Dans les secteurs critiques (aéronautique, énergie, automobile), la traçabilité et la conformité sont essentielles. L'IA contribue à améliorer la qualité et la sécurité des systèmes en garantissant un suivi complet des exigences, des tests et des anomalies. Selon Visure Solutions, l'IA offre des capacités avancées de traceability matrix et d'impact analysis, permettant de relier automatiquement chaque exigence à ses tests associés et à leurs résultats.

Selon Visure Solutions, l'IA offre des capacités avancées de traceability matrix et d'impact analysis, permettant de relier automatiquement chaque exigence à ses tests associés et à leurs résultats.

Safran, par exemple, utilise des algorithmes de corrélation IA pour valider les comportements moteurs sur des bancs d'essais physiques et numériques. Cette approche réduit les risques d'erreur humaine et renforce la fiabilité des certificats de conformité. Schneider, de son côté, déploie des modèles IA sur EcoStruxure Machine Expert pour identifier automatiquement les scénarios de test manquants et améliorer la couverture de validation avant mise en production.

Au-delà de la conformité, l'IA favorise la sécurité logicielle. Les systèmes d'analyse de code enrichis d'apprentissage automatique détectent les vulnérabilités de sécurité avant l'exécution, complétant le paradigme *Shift-Left* promu par VM Software House et Zuci Systems. Cela permet de déplacer la détection des anomalies vers les phases amont du développement, réduisant ainsi le coût des corrections.

Au-delà des exigences de conformité, l'intelligence artificielle contribue également à renforcer la sécurité logicielle. L'intégration de modèles d'apprentissage automatique dans les outils d'analyse de code permet désormais d'identifier les vulnérabilités dès les premières étapes du développement, dans l'esprit du paradigme *Shift-Left*. Cette approche déplace la détection des anomalies vers l'amont du cycle de vie applicatif, réduisant ainsi significativement le coût et le temps associés aux corrections.

Sur le plan de la gouvernance, il devient essentiel de privilégier des systèmes d'IA traçables, explicables et auditables. Ces principes, déjà bien établis dans des secteurs fortement régulés comme le médical, s'imposent progressivement comme des exigences de qualité dans le développement logiciel en cohérence avec les standards ISO/IEC 29119 relatifs aux tests des applications critiques.

6. Amélioration de l'expérience utilisateur & valeur client

Des processus plus rapides, plus fiables et plus intelligents se traduisent par une expérience utilisateur (interne ou externe) améliorée avec: moins d'erreurs, moins d'attente, plus de pertinence. Par exemple, le chatbot IA dans Opencell aide clients, métiers et consultants à obtenir des réponses plus rapidement, ce qui est une valeur ajoutée directe.

Dans *AI AND MACHINE LEARNING IN BUSINESS PROCESS AUTOMATION: INNOVATING WAYS*, les auteurs expliquent que l'IA dans l'automatisation des processus crée une meilleure interaction, une réactivité accrue, et une personnalisation des services [34].

Inconvénients et limites de l'IA dans les tests logiciels

1. Coûts initiaux, dépendance et besoin accru de compétences spécialisées

Si l'IA représente une avancée majeure pour l'automatisation des tests, sa mise en œuvre demeure coûteuse et complexe. L'un des premiers obstacles identifiés par la littérature et confirmé par les entreprises étudiées est la barrière d'entrée financière et technique liée à la formation des équipes et à l'intégration de nouvelles infrastructures.

Les retours d'expérience montrent qu'une proportion importante de projets d'intelligence artificielle près de quatre sur dix n'atteignent pas la phase de déploiement opérationnel. Comme le soulignent Mikalef et al. (2024) dans leur étude sur le déploiement de l'IA au sein de l'initiative *Digital-First Workforce* de Johnson & Johnson, ces difficultés ne proviennent pas principalement d'un manque de données, mais plutôt d'une insuffisance de compétences internes pour comprendre, interpréter et maintenir les modèles. Cette observation met en évidence que la réussite d'un projet d'IA repose autant sur la maîtrise technologique que sur la capacité organisationnelle à intégrer durablement les solutions dans les processus métiers [35].

Les résultats de l'enquête LinkedIn corroborent cette tendance, avec 39 % des répondants citent le manque de compétences comme principal frein à l'adoption de l'IA dans les tests logiciels, suivi de 34 % qui évoquent les difficultés d'intégration avec les outils existants. Cette lacune engendre une dépendance accrue vis-à-vis des fournisseurs de solutions IA ou des prestataires externes, créant un risque de verrouillage technologique.

Les études de cas renforcent ce constat; chez Schneider Electric, l'intégration de modèles prédictifs au sein d'EcoStruxure a nécessité la création d'une cellule interne d'experts en data science et DevOps, augmentant temporairement les coûts opérationnels. De même, Opencell a dû investir dans des ressources NLP et en annotation de données textuelles pour garantir la fiabilité des scénarios de test générés automatiquement. Ces

investissements initiaux sont souvent amortis à moyen terme, mais ils constituent un frein organisationnel pour les structures de taille moyenne.

Enfin, la dépendance aux outils propriétaires (Testim, Functionize, GPT, Copilot, etc.) limite parfois la flexibilité technologique. Les modèles fermés rendent difficile l'adaptation fine des algorithmes ou l'audit des résultats, une contrainte particulièrement sensible pour les secteurs soumis à des exigences réglementaires strictes (aéronautique, défense, santé).

2. Retour sur investissement incertain

L'investissement initial nécessaire à la mise en œuvre de solutions d'intelligence artificielle qu'il s'agisse des licences, de l'infrastructure, de la formation ou de la collecte de données demeure souvent conséquent, en particulier pour les petites et moyennes entreprises. Ce constat rejoint les limites fréquemment évoquées lors des entretiens, notamment en matière de coûts et de contraintes budgétaires.

De nombreux projets d'IA peinent à atteindre les résultats escomptés lorsqu'ils sont mal calibrés ou insuffisamment pilotés, traduisant le phénomène bien connu de *«overpromise versus underdeliver»*. Par exemple, la revue *Artificial Intelligence and Business Value: A Literature Review* identifie explicitement les obstacles financiers parmi les principaux facteurs inhibant la création de valeur issue de l'IA [36].

3. Fiabilité, biais, explicabilité et confiance

Les modèles d'IA peuvent produire des erreurs, des biais, des faux positifs/négatifs. Sans explicabilité, leur usage dans des processus critiques devient risqué. Les participants du questionnaire l'ont justement mentionné comme une amélioration souhaitée («meilleure explicabilité», «auditabilité»).

Des travaux montrent que même les algorithmes Deep Learning ne surpassent pas toujours les modèles plus simples lorsqu'ils sont appliqués à des données structurées ce qui tempère l'enthousiasme pour l'IA comme solution universelle. *Deep Learning in Business Analytics* discute de cette «clash of expectations» [37].

4. Risques organisationnels, éthiques et de dépendance excessive

Une dépendance excessive à l'intelligence artificielle peut, à terme, entraîner une érosion progressive des compétences humaines et une réduction de l'autonomie collective des équipes. Cette évolution soulève plusieurs enjeux éthiques majeurs, en particulier en matière de protection des données à caractère personnel, les biais des modèles et la responsabilité des décisions automatisées et de justice algorithmique.

À cet égard, l'étude *Artificial Intelligence as a Driver of Business Process Transformation* met en garde contre les dérives possibles d'une transformation imposée sans alignement stratégique. Les auteurs y rappellent que l'intelligence artificielle doit être envisagée comme un levier d'amélioration au service des processus métiers, et non comme une contrainte technologique dictant la transformation organisationnelle [38].

5. Qualité et gouvernance des données: un talon d'Achille

L'efficacité des modèles d'IA dépend étroitement de la qualité des données utilisées pour leur apprentissage. Or les environnements de test contiennent souvent des informations incomplètes, biaisées ou bruitées, ce qui peut fausser les prédictions et réduire la fiabilité des verdicts.

Zuci Systems souligne que le risque de biais est particulièrement élevé lorsque les modèles sont alimentés par des historiques de tests comportant des "flaky tests" c'est-à-dire des tests instables dont les résultats varient sans modification du code. Ces fluctuations peuvent tromper les algorithmes de classification et générer des faux positifs ou négatifs.

Dans le cas de Safran, les équipes de validation ont constaté que les modèles d'IA de détection d'anomalies sur les données moteur nécessitaient une curation manuelle intensive avant d'atteindre un niveau acceptable de précision. Le processus d'étiquetage et de nettoyage des données s'est révélé aussi crucial que chronophage.

De manière plus générale, la gouvernance des données de test incluant la traçabilité, le versionnage et l'auditabilité constitue une condition essentielle pour assurer la fiabilité et la reproductibilité des résultats produits par les modèles. Sans cette rigueur, les systèmes risquent d'amplifier les biais historiques ou d'extrapoler des comportements erronés.

Les entreprises doivent également veiller à la confidentialité et à la sécurité des données utilisées pour l'entraînement des modèles. Les environnements de test peuvent contenir des informations sensibles telles que des identifiants, des configurations de systèmes industriels ou des journaux d'activité. Un usage non contrôlé de l'IA, notamment via des services ou API externes, peut introduire des risques de fuite ou de non-conformité réglementaire.

Sans gouvernance explicite, les bénéfices technologiques apportés par l'intelligence artificielle peuvent être compromis par des risques systémiques liés à la donnée, à la sécurité et à la confiance.

6. Manque d'explicabilité et nouveaux risques éthiques

L'un des principaux paradoxes liés à l'intelligence artificielle appliquée au test logiciel est que plus les systèmes deviennent performants, plus leur fonctionnement devient opaque. La plupart des outils d'automatisation fondés sur l'IA agissent comme des "boîtes noires", produisant des verdicts de test sans expliciter la logique qui a conduit à ces conclusions.

Cette opacité soulève une question essentielle: comment justifier un résultat si le modèle ne permet pas de comprendre pourquoi une anomalie a été détectée ou ignorée? Ce manque d'explicabilité constitue un véritable défi méthodologique, car il remet en cause la fiabilité du verdict et complique la validation du processus de test.

Dans les environnements critiques, cette absence de transparence entre en conflit avec les exigences de certification et d'audit, qui imposent une traçabilité complète des processus de validation. Les organisations soumises à des normes strictes, telles que celles du secteur aéronautique, doivent donc compléter les systèmes d'analyse automatique par un contrôle humain renforcé afin de garantir la conformité et la responsabilité des décisions.

Un autre risque est celui d'un "effet tunnel algorithmique", où les équipes finissent par accorder une confiance aveugle aux recommandations de l'IA, au détriment du jugement d'expert. Cette dérive cognitive peut entraîner une dépendance excessive aux outils, réduisant la vigilance et la capacité d'analyse critique des testeurs.

Les retours du terrain montrent d'ailleurs que la confiance dans l'IA repose davantage sur la perception de la sécurité et de la fiabilité des résultats que sur la compréhension réelle de leur production. L'explicabilité reste souvent reléguée au second plan, tant que les performances semblent satisfaisantes. Or, cette confiance implicite demeure fragile : un seul incident majeur, comme un faux négatif dans un test critique, peut suffire à la remettre en cause et à générer une défiance organisationnelle durable.

Pour pallier ces limites, l'intégration de mécanismes d'IA explicable (XAI) dans les chaînes de test apparaît comme une voie prometteuse. Ces approches visent à associer à chaque verdict une justification intelligible, facilitant la certification, la traçabilité et la supervision humaine. En combinant performance algorithmique et transparence, elles permettraient de restaurer un équilibre entre automatisation et responsabilité.

7. Complexité d'intégration et dette technique des modèles IA

L'introduction de l'intelligence artificielle dans un pipeline de test ne se résume pas à l'ajout d'un outil supplémentaire car elle impose souvent une refonte profonde de l'écosystème CI/CD, des flux de données et des métriques de qualité. Cette complexité d'intégration peut paradoxalement ralentir les premiers cycles de développement, surtout lorsque les systèmes existants reposent sur des architectures hétérogènes ou des processus fortement couplés.

Dans certains cas, la mise en place de tests automatisés entre plusieurs applications d'entreprise (par exemple PLM, ERP et CRM) nécessite une orchestration complexe via des plateformes cloud et des connecteurs API. Chaque ajout de composant IA doit alors être soigneusement aligné avec les standards internes de sécurité, de conformité et de performance, afin d'éviter les ruptures dans la chaîne d'intégration continue.

Par ailleurs, l'usage de l'IA introduit une nouvelle forme de dette technique celle des modèles eux-mêmes. Comme le code source, un modèle d'apprentissage automatique vieillit, se dégrade ou dérive lorsque son environnement change. Sans réentraînement

régulier, il perd en précision et peut produire des résultats incohérents, voire erronés. Ce phénomène de dérive du modèle, ou *model drift*, est particulièrement critique dans les contextes dynamiques où les pratiques de développement et les jeux de données évoluent rapidement.

Cette dette vient s'ajouter à celle déjà existante dans les pipelines de tests instables (*flaky tests*), dépendances obsolètes, scripts non maintenus, amplifiant la complexité opérationnelle. Pour la maîtriser, les entreprises doivent adopter des stratégies efficaces de maintenance continue des modèles suivi des performances, recalibrage périodique, actualisation des jeux de données et audit des comportements algorithmiques.

Sans ces mécanismes de gouvernance et de supervision, les bénéfices initiaux de l'IA risquent de s'inverser. Une solution mal calibrée peut en effet introduire de nouveaux faux positifs, ralentir les campagnes de test et fausser les indicateurs de qualité, transformant ainsi un levier d'optimisation en source potentielle d'instabilité.

8. Flakiness algorithmique et risques d'automatisation excessive

L'un des risques les plus sous-estimés est la flakiness induite par l'IA elle-même. En cherchant à corriger des comportements instables, certains modèles peuvent introduire une variabilité de décision liée à leur nature probabiliste. Les tests deviennent alors non déterministes car une exécution identique peut produire des résultats différents selon la sensibilité du modèle**.**

Cette problématique rejoint la notion de "flaky tests" étudiée par Parry et al le non-déterminisme, même minime, remet en cause la confiance dans le verdict de test.

De plus, l'automatisation excessive peut dégrader la vigilance humaine. Les testeurs deviennent de simples superviseurs de pipeline, sans nécessairement comprendre les décisions des modèles. Dans l'étude de cas Opencell, les ingénieurs ont noté que la génération automatique de scénarios réduisait la réflexion critique sur la pertinence métier des tests, un effet secondaire bien connu dans les approches hyper-automatisées.

La sur-automatisation peut provoquer un effet tunnel: les modèles IA, entraînés sur des patterns existants, peuvent reproduire les angles morts historiques et ignorer des scénarios atypiques. Cela renforce le besoin d'une supervision humaine diversifiée et d'une hybridation entre heuristiques expertes et apprentissage automatique.

Discussion critique et conclusion

L'IA dans les processus est un outil puissant, mais pas une panacée universelle. Son succès dépend fortement des conditions contextuelles qui sont: la qualité des données, alignement stratégique, capacités organisationnelles, gouvernance. L'écart entre les promesses

académiques et la réalité opérationnelle persiste, notamment en termes de fiabilité, d'explicabilité et de retour sur investissement.

Une adoption responsable exige une approche humaine-centered, où les utilisateurs, experts et décideurs co-construisent les systèmes (frameworks comme celui de *Responsible AI Implementation*).

À la lumière de tes études de cas et des interviews, on voit que les organisations doivent anticiper non seulement les bénéfices mais aussi les effets secondaires (dépendance excessive, biais, dette technique). Enfin, l'IA ne doit pas remplacer les compétences humaines mais les augmenter: l'enjeu est de trouver l'équilibre entre automatisation et supervision humaine.

En conclusion, l'intégration de l'IA dans les processus (tests ou autres) offre des bénéfices considérables: efficience, prise de décision augmentée, agilité, innovation et meilleure expérience. Mais ces bénéfices viennent avec des coûts et des risques: coûts d'entrée élevés, manque de compétences, défi de la confiance, complexité d'intégration et enjeux éthiques.
La valeur de l'IA se concrétise lorsqu'elle est inscrite dans une gouvernance robuste, un cadre explicable et responsable, et un accompagnement humain fort.

Cette analyse sert de fondement au pour la suite de cette rédaction, où seront élaborées des recommandations stratégiques pour les entreprises souhaitant faire entrer l'IA dans leurs processus de façon pragmatique et durable.

CHAPITRE6: SYNTHÈSE ET RECOMMANDATIONS

Synthèse

L'intégration de l'intelligence artificielle dans les processus de test logiciel marque une évolution majeure de l'ingénierie logicielle contemporaine. Cette recherche a permis de mettre en évidence, à travers une approche croisée combinant études de cas (Sparco, Schneider Electric, Safran, Opencell) et enquête empirique auprès de professionnels les bénéfices, les défis et les prérequis de succès de cette transformation.

Globalement, l'IA s'impose comme un levier d'optimisation du cycle de développement, en particulier dans les contextes où les volumes de tests, la complexité des systèmes et les exigences de qualité rendent les approches manuelles ou semi-automatisées obsolètes. Les bénéfices observés dans les entreprises étudiées se traduisent par une réduction des temps de test de 40 à 70 %, une augmentation significative de la couverture fonctionnelle, et une meilleure stabilité des pipelines CI/CD grâce à l'introduction de modèles prédictifs et de mécanismes *self-healing*.

Chez Schneider Electric, par exemple, les campagnes de tests de régression sont passées de 72 heures à moins de 24 heures après l'adoption de modèles d'apprentissage supervisé pour la priorisation des tests. Safran, de son côté, a démontré l'impact de l'IA sur l'analyse prédictive dans des environnements critiques, réduisant drastiquement les temps d'inspection des données moteurs.

Enfin, chez Opencell et Sparco, l'utilisation du traitement du langage naturel (NLP) et de l'IA générative (GPT, Codex) a permis de transformer la manière dont les cas de test sont conçus, documentés et reliés aux exigences métiers.

Ces résultats sont cohérents avec les retours des 41 répondants à l'enquête LinkedIn: près de 49 % connaissent déjà l'usage de l'IA dans les tests, et 22 % l'ont appliquée directement dans leur entreprise. Les bénéfices les plus cités concernent l'amélioration de la qualité logicielle (34 %), le gain de temps (22 %), et la meilleure couverture de test (22 %). Cependant, les professionnels soulignent également les limites: manque de compétences internes (39 %), difficulté d'intégration (34 %) et problèmes de fiabilité des modèles (24 %). Ces chiffres traduisent un écart entre la maturité technologique et la maturité organisationnelle, qui conditionne le succès de l'adoption.

Ainsi, si les entreprises les plus avancées (comme Schneider ou Safran) ont réussi à industrialiser les tests intelligents en les intégrant dans des architectures cloud et DevOps, d'autres restent freinées par des obstacles liés à la culture, aux compétences et à la gouvernance des données.

La principale conclusion de cette étude est donc que l'IA n'est pas un substitut au testeur humain, mais un amplificateur de sa capacité d'analyse et d'anticipation. Elle introduit une logique de test prédictif et auto-adaptatif, qui transforme la validation logicielle en un processus continu, intelligent et collaboratif.

Cependant, cette transformation n'est durable que si elle repose sur trois fondations: la maîtrise technique, la gouvernance de la donnée, et la maturité organisationnelle.

En somme, l'IA permet de switcher d'un modèle de test réactif vers un modèle proactif et cognitif, où les erreurs ne sont plus seulement détectées, mais anticipées et prévenues. Toutefois, les risques de dérive (opacité, biais, dépendance, dérive des modèles) rappellent la nécessité d'une approche hybride où l'humain conserve un rôle central dans la supervision, l'interprétation et la décision.

Recommandations

a) Recommandations techniques

Les recommandations techniques qui suivent visent à accompagner les entreprises dans la mise en marche opérationnelle de l'intelligence artificielle dans leurs scénarios de test, en s'appuyant sur les constats empiriques de cette recherche.

1. Favoriser une intégration progressive et modulaire

L'adoption de l'IA doit être envisagée comme un processus incrémental. Les entreprises devraient commencer par des cas d'usage ciblés et mesurables: priorisation des tests, génération automatique de scénarios ou analyse de logs par apprentissage supervisé. Cette approche modulaire permet de valider la performance du modèle sur un périmètre restreint avant d'étendre son champ d'application.

À titre d'exemple, Schneider Electric a initié l'usage de l'IA par la seule priorisation des tests CI/CD avant d'introduire la génération de scripts automatisés. Ce phasage a permis de réduire les risques d'échec et d'adapter les modèles au contexte interne.

2. Assurer la qualité et la gouvernance des données de test

La fiabilité des résultats dépend de la qualité des données d'entraînement. Il est donc essentiel d'instaurer une gouvernance des données de test fondée sur la traçabilité, le versionnage et la documentation. Les entreprises doivent notamment:

- Centraliser les données de test dans des référentiels sécurisés (Data Lakes, MLOps).
- Définir des métriques de qualité (taux de bruit, biais, couverture).
- Anonymiser les données sensibles pour garantir la conformité RGPD.
- Implémenter des politiques de *data lineage* pour assurer la reproductibilité. Cette rigueur est indispensable pour prévenir les biais de modèles, les falsifications de résultats et les dérives statistiques.

3. Intégrer des mécanismes d'explicabilité et de contrôle humain

La mise en place de l'IA dans les tests logiciels doit s'accompagner de solutions d'IA explicable (XAI) permettant de comprendre les décisions algorithmiques. Les outils doivent offrir des rapports interprétables:

- Justification des verdicts de test,
- Corrélation entre les entrées et les décisions,
- Indicateurs de confiance des modèles.

Cette transparence est particulièrement cruciale dans les domaines critiques (aéronautique, santé, énergie), où chaque décision doit être auditée. L'IA doit donc rester sous supervision humaine, garantissant un équilibre entre automatisation et responsabilité.

4. Mettre en place une maintenance continue des modèles (ModelOps)

Les modèles d'IA se dégradent avec le temps. Pour éviter cette perte de performance, les entreprises doivent instaurer des mécanismes de surveillance et de recalibrage automatiques:

- Monitoring en continu des performances (précision, taux de faux positifs/négatifs).
- Réentraînement périodique avec de nouvelles données.
- A/B testing entre versions de modèles.
- Documentation systématique des ajustements.

Cette approche garantit la stabilité du pipeline de test et la cohérence entre les versions de code, d'environnement et de modèle.

5. Promouvoir l'hybridation des approches

Plutôt que de tout confier à des modèles prédictifs, il est recommandé de combiner l'IA avec des règles expertes ou heuristiques métier. Cette hybridation permet de bénéficier à la fois de la puissance d'analyse des modèles et de l'expertise humaine accumulée.

Chez Opencell et Sparco, par exemple, l'IA générative (pour la création de tests Gherkin) a été complétée par des validations manuelles assurant la cohérence métier.

De plus, l'introduction de l'IA dans le test doit s'inscrire dans une logique DevOps/MLOps, favorisant une boucle d'amélioration continue où chaque anomalie détectée enrichit le modèle.

b) Recommandations organisationnelles

Les bénéfices techniques de l'IA ne peuvent se concrétiser sans un changement de paradigme organisationnel. Les entreprises doivent adapter leur culture, leurs processus et leurs compétences pour tirer pleinement parti de l'automatisation intelligente.

1. Développer une culture de l'expérimentation et de la collaboration

L'adoption de l'IA dans le test nécessite une culture d'apprentissage continu. Les équipes doivent être encouragées à expérimenter, évaluer et partager leurs retours d'expérience.

Cette démarche suppose:

- Des espaces de collaboration inter-fonctionnelle (Dev–QA–Ops–Data).
- Des formations régulières à la compréhension et à la supervision des modèles.
- Une communication transparente autour des résultats et limites de l'IA.

Les organisations comme Schneider ont d'ailleurs mis en place des "AI Labs" internes, où data scientists et testeurs travaillent conjointement sur des prototypes avant leur industrialisation.

2. Repenser les rôles et compétences

Le métier de testeur évolue vers un rôle de superviseur de modèles IA. Les compétences attendues dépassent la simple maîtrise des outils de test: il s'agit désormais de comprendre les bases du machine learning, des biais algorithmiques et des architectures de données.

Les entreprises doivent donc investir dans des formations ciblées:

- Compréhension des modèles IA et des métriques de performance.
- Lecture et interprétation des rapports d'explicabilité.

- Gouvernance de la donnée et conformité réglementaire.

Ces nouvelles compétences favorisent une responsabilisation accrue des équipes QA et renforcent leur position stratégique dans la chaîne de valeur numérique.

3. Instaurer une gouvernance éthique et responsable de l'IA

Au-delà des aspects techniques, la gouvernance éthique devient un pilier central. L'utilisation de l'IA dans des processus critiques impose une responsabilité partagée entre les développeurs, les testeurs et la direction.

Les entreprises doivent adopter une charte interne d'IA responsable, s'inspirant des cadres de référence européens (AI Act) ou internationaux (ISO/IEC 42001).

Cette gouvernance doit garantir:

- La traçabilité des décisions IA,
- L'évaluation des impacts sur l'emploi et les compétences,
- L'application des principes de transparence et d'équité.

Une telle approche prévient les dérives d'automatisation aveugle et renforce la confiance des parties prenantes.

4. Aligner l'IA avec la stratégie globale d'innovation

Enfin, l'intégration de l'intelligence artificielle dans les tests ne doit pas être perçue comme un projet isolé, mais comme un levier stratégique d'innovation et de compétitivité.

Les directions doivent aligner les initiatives IA avec les priorités métiers: réduction du time-to-market, amélioration de la qualité, durabilité et sécurité des systèmes.

Chez Sparco, cette approche intégrée s'est traduite par une synergie entre les projets IA, PLM et ERP, soutenant la transformation digitale globale de l'entreprise.

CHAPITRE 7: LIMITES DE MA RECHERCHE ET PERSPECTIVES FUTURES

Limites

Bien que ce travail ait permis d'explorer en profondeur les apports et les limites de l'intelligence artificielle dans les tests logiciels, il convient de reconnaître qu'il comporte plusieurs limites méthodologiques et empiriques.

Tout d'abord, la principale contrainte tient à la taille de l'échantillon et à la portée de la collecte de données. Les entretiens et questionnaires diffusés via LinkedIn ont permis de recueillir les avis de presque 50 professionnels issus de divers secteurs, mais ce nombre

reste relativement modeste pour tirer des conclusions pleinement généralisables. Une participation plus large, couvrant davantage de pays, d'industries et de profils (data scientists, test managers, ingénieurs QA seniors, responsables R&D) aurait permis d'obtenir une vision plus représentative des pratiques et des perceptions liées à l'IA dans les tests logiciels.

De même, le choix de se concentrer sur quatre études de cas spécifiques (Sparco, Safran, Schneider Electric et Opencell) a permis d'approfondir la compréhension des contextes d'implémentation, mais limite la diversité des environnements observés. D'autres secteurs comme la santé, la finance ou les télécommunications auraient pu enrichir l'analyse en apportant des contraintes et des dynamiques différentes, notamment sur le plan réglementaire, éthique ou opérationnel.

Sur le plan scientifique, cette recherche a volontairement adopté une approche qualitative et exploratoire. Si elle a permis d'identifier les tendances dominantes et les facteurs clés de succès, elle n'a pas mobilisé d'analyse quantitative avancée (modélisation statistique, corrélations entre variables, ou indicateurs de performance standardisés). Une telle démarche aurait nécessité un accès plus large à des bases de données industrielles ou à des résultats de projets pilotes, ce qui dépasse le cadre de ce travail. De plus, certaines notions comme la maturité organisationnelle de l'IA, la gouvernance des données ou l'explicabilité des modèles auraient gagné à être mesurées à l'aide de grilles d'évaluation normalisées ou d'indicateurs de performance clés. Ces éléments auraient permis de mieux comparer les entreprises entre elles et de proposer un modèle d'évaluation reproductible.

Perspectives futures

Les perspectives futures s'articulent autour de deux axes principaux: le prolongement scientifique de cette recherche et les implications pratiques pour les entreprises.

Sur le plan académique, un prolongement naturel de ce travail consisterait à approfondir la dimension empirique en élargissant le périmètre d'observation. Des études longitudinales, suivant l'évolution des pratiques d'IA dans le temps, permettraient de mesurer la durabilité des gains observés et d'évaluer la manière dont les organisations maintiennent la performance de leurs modèles (réentraînement, dérive, obsolescence).

De même, l'utilisation de méthodes mixtes combinant enquêtes quantitatives à grande échelle, entretiens semi-directifs et analyses statistiques offrirait une vision plus équilibrée et robuste.

Du point de vue technologique, plusieurs pistes méritent d'être explorées. D'abord, l'adoption de l'intelligence artificielle explicable (*Explainable AI*) dans les plateformes de test représente un chantier prometteur: il s'agira de concevoir des systèmes capables non seulement de détecter ou d'anticiper des anomalies, mais aussi d'en justifier la logique décisionnelle. Ensuite, le développement de solutions d'IA fédérée, où les modèles apprennent à partir de données distribuées sans les centraliser, pourrait répondre aux

enjeux croissants de confidentialité et de conformité réglementaire (notamment le RGPD européen).

Sur le plan organisationnel, il conviendrait d'analyser plus en profondeur l'évolution du rôle humain dans cette transformation. Si l'intelligence artificielle automatise les tâches répétitives, routinières et prédictives, la valeur ajoutée des testeurs se déplacera vers des activités d'interprétation, de supervision et d'éthique des systèmes. Les futurs travaux pourraient ainsi explorer comment les équipes QA se forment, collaborent et redéfinissent leurs responsabilités dans un environnement où les décisions deviennent partiellement automatisées.

CONCLUSION

L'adoption de l'intelligence artificielle (IA) au sein des processus de test logiciel représente un tournant important dans le développement des pratiques d'ingénierie et de développement. Ce mémoire s'est attaché à comprendre en profondeur comment et dans quelles conditions l'IA transforme le test logiciel, en explorant à la fois ses bénéfices, ses limites et ses implications organisationnelles.

À travers une approche fondée sur la revue de littérature, l'analyse de cas concrets (Sparco, Schneider Electric, Safran, Opencell) et l'enquête menée auprès de professionnels, il a été possible de dresser une vision complète et équilibrée du rôle croissant de l'intelligence artificielle dans les activités de qualité et de validation logicielle.

Synthèse des résultats et constats majeurs

Les résultats de cette recherche montrent clairement que l'IA agit comme un levier de performance et de fiabilité pour les processus de test. Dans un contexte où les entreprises doivent livrer rapidement des solutions logicielles complexes, souvent interconnectées à d'autres systèmes, l'intelligence artificielle apporte une réponse à la fois technique et stratégique aux défis contemporains: réduction des délais, amélioration de la précision et meilleure gestion de la complexité.

Les études menées démontrent que l'IA permet de diminuer significativement la durée d'exécution des tests, d'augmenter la couverture fonctionnelle, et d'améliorer la détection précoce des anomalies.

Chez Schneider Electric, par exemple, la priorisation intelligente des tests via l'apprentissage automatique a permis une réduction de 60 % du temps des campagnes de régression, tout en améliorant la stabilité des pipelines CI/CD. Safran, de son côté, a utilisé des algorithmes prédictifs pour analyser les données moteurs en seulement sept jours au lieu de soixante, illustrant la possibilité de l'IA à traiter et corréler des volumes massifs de données techniques.

Dans le cas d'Opencell, l'automatisation de la rédaction des scénarios Gherkin à partir des tickets Jira grâce au NLP a permis de raccourcir les cycles de test tout en garantissant la cohérence avec les exigences métiers. Enfin, Sparco a intégré des outils IA dans ses flux Salesforce–ERP–AWS, améliorant la supervision et la qualité des échanges de données

Ces expériences convergent vers un constat central: l'intelligence artificielle ne se contente pas d'automatiser, elle apprend et s'adapte. En d'autres termes, elle introduit une dynamique nouvelle dans le test logiciel, où les campagnes deviennent évolutives, les scripts se réparent d'eux-mêmes (self-healing), et les priorisations reposent sur des modèles statistiques capables d'anticiper les zones de risque.

Cette transformation place le test au cœur d'un processus d'amélioration continue, où la donnée devient un actif stratégique et la machine un partenaire d'aide à la décision.

L'enquête menée auprès de 41 professionnels sur LinkedIn renforce ces conclusions empiriques. Les répondants soulignent que l'IA améliore la qualité logicielle (34 %), accélère les délais (22 %), et augmente la couverture de test (22 %). Les avis convergent aussi sur la nécessité de renforcer la sécurité, la fiabilité et l'explicabilité des outils intelligents. Ces résultats montrent que les bénéfices sont bien identifiés et mesurables, mais qu'ils s'accompagnent d'une exigence croissante en matière de contrôle et de compétence humaine.

Réponse à la problématique

La problématique centrale de ce mémoire interrogeait la valeur réelle de l'intelligence artificielle dans les tests logiciels:

Quels sont les avantages et les inconvénients de l'intelligence artificielle dans les processus de test et d'automatisation, et dans quelle mesure cette technologie contribue-t-elle à améliorer la qualité et la performance organisationnelle?

Les analyses menées permettent d'apporter une réponse nuancée. D'un point de vue fonctionnel, l'IA représente une avancée majeure: elle accélère le time-to-market, réduit les coûts, et accroît la robustesse des logiciels testés. Elle permet aux entreprises d'atteindre un niveau d'agilité inédit, en testant en continu des systèmes de plus en plus distribués et en tirant parti d'une automatisation intelligente.

D’un point de vue stratégique, elle favorise une meilleure alignement entre les exigences métiers et les activités techniques, notamment grâce aux outils de génération automatique de scénarios ou à la traçabilité assistée par IA.

Cependant, l’étude révèle également les limites intrinsèques à cette transformation. Les résultats ne sont probants que si certaines conditions sont réunies: qualité et quantité suffisante de données, maîtrise des modèles algorithmiques, infrastructure adaptée et, surtout, présence d’équipes capables d’interpréter et de superviser les résultats.

Le questionnaire a montré que le manque de compétences internes (39 %) et la difficulté d’intégration (34 %) restent les freins principaux. L’IA est donc un atout, mais elle n’est pas une solution immédiate car elle exige un accompagnement humain, méthodologique et culturel.

On peut donc conclure que En somme, l’intelligence artificielle ne remplace pas le testeur, elle redéfinit sa mission. Le rôle du professionnel évolue vers la supervision, l’interprétation et la gouvernance de systèmes intelligents.

Enseignement global et portée de la recherche

Au-delà des résultats chiffrés, cette recherche a permis de mettre en évidence un changement plus profond qui est celui du test comme fonction cognitive et stratégique.

Autrefois perçu comme une simple étape de validation en fin de cycle, le test devient désormais un processus apprenant, intégré dans la chaîne de valeur du développement. Grâce à l’IA, le test logiciel est capable de s’adapter, de prédire, de corriger et même d’expliquer du moins partiellement ses décisions.

Les modèles d’intelligence artificielle introduisent ainsi une nouvelle intelligence du test, où la prévention remplace la détection et où la correction devient proactive. Ce changement paradigmatique bouleverse l’équilibre entre humains et machines: les testeurs deviennent des superviseurs de flux d’information et des garants de la fiabilité algorithmique.

Par ailleurs, ce travail a mis en lumière la dimension organisationnelle de la réussite d’un projet IA. L’intégration de modèles d’apprentissage dans les pipelines CI/CD n’est pas seulement une question technique, mais une transformation culturelle et structurelleLes entreprises les mieux positionnées sont celles qui ont été capables d’instaurer un environnement collaboratif entre data scientists, développeurs et ingénieurs QA. L’intelligence collective devient alors un moteur aussi essentiel que l’intelligence artificielle elle-même.

Enfin, cette recherche montre que l’IA pose des questions de gouvernance, de responsabilité et d’éthique. Le manque d’explicabilité des modèles, les risques de biais ou les erreurs de classification peuvent avoir des conséquences importantes sur la qualité finale du produit. Ces défis ne relèvent pas uniquement de la technologie, mais d’une approche globale combinant transparence, contrôle humain et alignement réglementaire.

Ainsi, si l'IA est un levier de performance, elle n'a de valeur que dans un cadre où la rigueur scientifique et la supervision humaine demeurent au centre du processus.

En conclusion, cette recherche montre que l'intelligence artificielle transforme durablement les tests logiciels, mais que sa réussite dépend de trois conditions fondamentales:

- La qualité des données et la maîtrise des environnements techniques;
- La compétence humaine et la capacité d'interprétation des résultats;
- La gouvernance éthique garantissant la transparence, la sécurité et la responsabilité.

Les entreprises étudiées qu'il s'agisse de Sparco, Safran, Schneider Electric ou Opencell ont démontré que lorsque ces conditions sont réunies, l'IA devient un catalyseur puissant d'innovation et de qualité. Elle permet d'allier rapidité, précision et traçabilité, tout en libérant les testeurs des tâches répétitives pour les recentrer sur l'analyse et la supervision.

Toutefois, cette transformation n'est pas exempte de risques. La dépendance aux outils propriétaires, la dette technique liée aux modèles et la complexité d'intégration exigent une vigilance permanente. L'intelligence artificielle doit être perçue non pas comme une finalité, mais comme un moyen au service d'un objectif humain: la fiabilité du logiciel et la confiance numérique

En définitive, ce mémoire aboutit à une conviction forte:

L'avenir du test logiciel n'est pas l'automatisation totale, mais la collaboration intelligente entre l'humain et la machine.

L'intelligence artificielle ne remplace pas la rigueur humaine, elle l'étend. Elle n'élimine pas l'erreur, mais aide à la comprendre et à l'anticiper. Elle ne réduit pas la responsabilité, mais la redéfinit.

C'est dans cette cohabitation, faite de complémentarité et de discernement, que se dessine la véritable révolution du test logiciel moderne.

RÉSUMÉ DU MÉMOIRE

Ce mémoire étudie l'intégration de l'intelligence artificielle dans les processus de test logiciel, un domaine en pleine mutation au croisement de l'ingénierie, de la data science et du management technologique. Face à la complexité croissante des systèmes numériques et à l'accélération des cycles de développement, les organisations recherchent aujourd'hui des méthodes capables d'assurer la qualité, la rapidité et la fiabilité des logiciels tout en optimisant les coûts.

L'objectif principal de cette recherche est d'analyser de manière critique les apports et les limites de l'IA dans l'automatisation des tests, en évaluant son impact sur la qualité logicielle, la productivité et la transformation organisationnelle. La méthodologie adoptée repose sur une approche mixte combinant revue de littérature académique, études de cas d'entreprises (Sparco, Schneider Electric, Safran, Opencell) et enquête empirique menée auprès de quarante et un professionnels du secteur via un questionnaire diffusé sur LinkedIn.

Les résultats révèlent que l'IA permet une réduction significative des cycles de test, une amélioration de la couverture fonctionnelle et une détection plus précoce des anomalies, tout en favorisant l'intégration continue (CI/CD) et la maintenance prédictive. Toutefois, ces bénéfices s'accompagnent de contraintes majeures: coûts d'implémentation élevés, manque de compétences internes, fiabilité parfois limitée des modèles et problèmes d'explicabilité. L'étude met ainsi en lumière le paradoxe d'une technologie à fort potentiel, mais dépendante de la maturité technique, des données et de la gouvernance organisationnelle.

En conclusion, l'intelligence artificielle ne remplace pas le testeur, elle en redéfinit le rôle. Elle transforme le test logiciel en un processus cognitif et adaptatif, où l'humain conserve une place centrale dans la supervision, l'analyse et la validation des résultats. Ce travail souligne ainsi que la valeur de l'IA réside moins dans son automatisation que dans sa capacité à renforcer la rigueur, la transparence et la fiabilité du développement logiciel moderne.

BIBLIOGRAPHIE

[1] J. O. Paul Ammann, «Introduction to softmare testing,». chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://api.pageplace.de/preview/DT0400.9781316774366_A29113267/preview-9781316774366_A29113267.pdf pp. 25-33, 2016.

[2] A. Bertolino, «Software Testing Research: Achievements, Challenges, Dreams» chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://people.scs.carleton.ca/~jeanpier/sharedF14/Bertolino-state-of-artinSWTesting.pdf pp. 11-16, 2007.

[3] N. Forsgren, «Accelerate-IT Revolution,» chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://itrevolution.com/wp-content/uploads/2022/06/ACC_Audio-Companion.pdf Portland, 2018.

[4] M. H. P. M. M. S. a. S. Y. Earl T. Barr, *The Oracle Problem in Software, chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://earlbarr.com/publications/testoracles.pdf* 2015.

[5] M. D. a. A. B. Anna Trudova, «Artificial Intelligence in Software Test Automation: A Systematic,» chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://www.scitepress.org/PublishedPapers/2020/94178/94178.pdf 2020.

[6] E. T. Barr, M. Harman e P. McMinn, «The Oracle Problem in Software Testing,» *https://ieeexplore.ieee.org/document/6963470,* 2015.

[7] R. P. M. Bagherzadeh, «Test Case Selection and Prioritization Using Machine Learning: A Systematic Literature Review,» *chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://arxiv.org/pdf/2106.13891,* 2021.

[8] S. B. M. F. panelVahid Garousi, «NLP-assisted software testing,» *https://www.sciencedirect.com/science/article/abs/pii/S0950584920300744?via%3Dihub,* 2020.

[9] G. OWAINPARRY, «ASurvey of Flaky Tests,» *chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://philmcminn.com/publications/parry2021.pdf?utm_source=chatgpt.com,* 2021.

[10] A. Sharif, «DeepOrder: Deep Learning for Test Case Prioritization in Continuous Integration Testing,» *https://arxiv.org/abs/2110.07443,* 2021.

[11] H. Sivaraman, «natural language processing for automated test case generation from software,» *https://www.researchgate.net/publication/389031328_NATURAL_LANGUAGE_PROCESSING_FOR_AUTOMATED_TEST_CASE_GENERATION_FROM_SOFTWARE_REQUIREMENT_DOCUMENTS,* 2019.

[12] Safran, «Leader in artificial intelligence for aerospace and defense».*https://www.safran-group.com/companies/safran-ai.*

[13] Safran, «Geospatial intelligence AI».*https://www.safran-group.com/products-services/geospatial-intelligence-ai.*

[14] M. Ignite, «Speed up testing by using Test Impact Analysis (TIA),» *https://learn.microsoft.com/en-us/azure/devops/pipelines/test/test-impact-analysis?view=azure-devops,* 2025.

[15] IEE, «A novel hybrid automatic intrusion detection system using machine learning technique for anomalous detection based on traffic prediction,» *https://ieeexplore.ieee.org/document/10127442,* 2023.

[16] S. Electric, «EcoStruxure™ Automation Expert».*https://www.se.com/us/en/product-range/23643079-ecostruxure-automation-expert/.*

[17] S. Electric, «The ultimate guide to open, software-centric industrial automation,» *chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://download.schneider-electric.com/files?p_Doc_Ref=998-22920153_EAE_eGuide_SI&p_enDocType=Brochure&p_File_Name=998-22920153_EAE_eGuide_for_System_Integrators_WEB.pdf,* 2023.

[18] M. Learn, «Speed up testing by using Test Impact Analysis (TIA),» *https://learn.microsoft.com/en-us/azure/devops/pipelines/test/test-impact-analysis?view=azure-devops,* 2025.

[19] smartmachines&factories, «How large language models can boost industrial automation,» *https://smartmachinesandfactories.co.uk/how-large-language-models-can-boost-industrial-automation/,* 2025.

[20] S. Electric, « ETEST User Guide,» *https://product-help.schneider-electric.com/Machine%20Expert/V2.0/en/ETEST/index.htm#t=ETEST%2FD-SE-0060866.html,* 2021.

[21] S. S.p.A, «Sparco DNA 100% passion,» *https://www.sparco-official.com/en/sparco-world,* 2021.

[22] M. Ignite, «Speed up testing by using Test Impact Analysis (TIA),» *https://learn.microsoft.com/en-us/azure/devops/pipelines/test/test-impact-analysis?view=azure-devops,* 2025.

[23] AWS, «Artificial Intelligence,» *https://aws.amazon.com/it/blogs/machine-learning/,* 2025.

[24] Opencell, «Opencell Platform».*https://www.opencellsoft.com/plateforme.*

[25] M. Ignite, «Speed up testing by using Test Impact Analysis (TIA),» *https://learn.microsoft.com/en-us/azure/devops/pipelines/test/test-impact-analysis?view=azure-devops,* 2025.

[26] Parasoft, «Guide for AI in software testing,» *https://fr.parasoft.com/white-paper/guide-for-ai-in-software-testing/,* 2025.

[27] Deduxer, «Avantages et inconvénients clés de l'IA dans le développement de logiciels,» *https://www.deduxer.studio/fr/blog/principaux-avantages-et-inconvenients-de-lintelligence-artificielle-pour-le-developpement-de-logiciels,* 2025.

[28] P. Reddy, «The Role of AI in Continuous Integration and Continuous Deployment (CI/CD) Pipelines: Enhancing Performance and Reliability,» *https://www.academia.edu/125793959/The_Role_of_AI_in_Continuous_Integration_and_Continuous_Deployment_CI_CD_Pipelines_Enhancing_Performance_and_Reliability?utm_source=chatgpt.com,* 2021.

[29] K. Veerappan, «L'intelligence artificielle (IA) dans les tests de logiciels».*https://www.zucisystems.com/be/blog/lintelligence-artificielle-dans-les-tests-de-logiciels/.*

[30] Visure, «L'IA dans les tests de logiciels».*https://visuresolutions.com/fr/alm-guide/ai-in-software-testing/.*

[31] V. Moskalenko, «Resilience and Resilient Systems of Artificial Intelligence,» *https://www.mdpi.com/1999-4893/16/3/165,* 2023.

[32] M. N. Tomasz Kluza, «L'IA dans l'analyse de code : avantages et défis,» *https://vmsoftwarehouse.fr/l-ia-dans-ianalyse-de-code-advantages-et-defis,* 2024.

[33] CloudBees, «DevOps test data platform doubles release velocity by saving 40K testing hours in one year».*https://www.cloudbees.com/customers/test-data-platform-doubles-release-velocity.*

[34] A. Rahman, «AI AND MACHINE LEARNING IN BUSINESS PROCESS AUTOMATION,» *https://nonhumanjournal.com/index.php/JMLDEDS/article/view/41,* 2024.

[35] O. Tona, «The Deployment of AI to Infer Employee Skills,» *https://onlinelibrary.wiley.com/doi/10.1111/isj.12594,* 2025.

[I. M. Enholm, «Artificial Intelligence and Business Value: a Literature Review,» 3 *https://www.researchgate.net/publication/354135976_Artificial_Intelligence_and_Busine* 6 *ss_Value_a_Literature_Review,* 2021.
]

[M. Schmitt, «Deep learning in business analytics: A clash of expectations and reality,» 3 *https://www.sciencedirect.com/science/article/pii/S2667096822000891,* 2023.
7
]

[N. Svetlana, «Artificial intelligence as a driver of business process transformation,» 3 *https://www.sciencedirect.com/science/article/pii/S1877050922017586,* 2022.
8
]