Testeur Certifié - Spécialiste

ISTQB®
Test d'application mobile
Niveau Fondation

Syllabus

Version 2019

Développé par International Software Quality Institute (iSQI)

International Software Testing Qualifications Board

Notice de copyright

Historique des révisions		
Version	Date	Remarques
Alpha	11 mai 2018	Version Alpha
Beta	27 janvier 2019	Version Beta
GA	28 mars 2019	Version GA
V2019	3 mai 2019	Version ISTQB® (publique)

Table des matières

Remerciements

Ce document a été produit par une équipe du groupe de travail Mobile Application Testing de l'International Software Testing Qualifications Board composé de Vipul Kocher (chair), Piotr Wicherski (vice-chair), José Diaz, Matthias Hamburg, Eran Kinsbruner, Björn Lemke, Samuel Ouko, Tal Pe'er, Ralf Pichler, Lloyd Roden, Nils Röttger, Angelina Samaroo, Yaron Tsubery.
Le groupe de travail remercie l'équipe de revue pour ses suggestions et ses commentaires.

Les personnes suivantes ont participé à l'examen, aux commentaires ou au vote de ce syllabus : Graham Bath, Veronica Belcher, Armin Born, Geza Bujdoso, YongKang Chen, Wim Decoutere, Frans Dijkman, Florian Fieber, David Frei, Péter Földházi Jr., Chaonian Guo, Attila Gyuri, Ma Haixia, Matthias Hamburg, Zsolt Hargitai, Hongbiao Liu, Ine Lutterman, Marton Matyas, Petr Neugebauer, Ingvar Nordström, Francisca Cano Ortiz, Nishan Portoyan, Meile Posthuma, Emilie Potin-Suau, Liang Ren, Lloyd Roden, Chaobo Shang, Mike Smith, Péter SÖtér, Marco Sogliani, Michael Stahl, Chris Van Bael, Paul Weymouth, Salinda Wickramasinghe, Minghui Xu.

Ce document a été officiellement publié par l'ISTQB® le 3 mai 2019.

0. Introduction

0.1 Objet de ce document

Ce syllabus constitue la base de la qualification des tests d'application mobile au Niveau Fondation. L'ISTQB® fournit ce syllabus comme suit :

1. Aux bureaux nationaux de l'ISTQB, pour le traduire dans leur langue locale et accréditer les prestataires de formation. Les bureaux nationaux de l'ISTQB peuvent adapter le syllabus à leurs besoins linguistiques particuliers et modifier les références pour s'adapter à leurs publications locales.
2. Aux organismes de certification, afin d'obtenir des questions d'examen dans leur langue locale adaptées aux objectifs d'apprentissage de chaque syllabus.
3. Aux organismes de formation, afin de produire des cours et déterminer les méthodes d'enseignement appropriées.
4. Aux candidats à la certification, afin de se préparer à l'examen (dans le cadre d'une formation ou de manière indépendante).
5. À la communauté internationale de l'ingénierie des logiciels et des systèmes, pour faire progresser la profession du test des logiciels et des systèmes, et comme base pour les livres et les articles.

L'ISTQB® peut permettre à d'autres entités d'utiliser ce syllabus à d'autres fins, sous condition qu'elles aient obtenu une autorisation écrite préalable.

0.2 La certification « test d'application mobile » de Niveau Fondation

La qualification de niveau Fondation s'adresse à toute personne impliquée dans le test logiciel qui souhaite élargir ses connaissances sur les tests d'applications mobiles ou toute personne qui souhaite commencer une carrière spécialisée dans le test d'applications mobiles.

Les informations sur les tests d'applications mobiles décrites dans le syllabus ISTQB® Testeur Certifié de Niveau Fondation [ISTQB CTFL 2018] ont été prises en compte lors de la création de ce syllabus.

0.3 Objectifs Métier

Cette section énumère les objectifs-métier attendus d'un candidat qui a obtenu la certification de test d'application mobile de niveau fondation.

MAT-01 Comprendre et revoir les objectifs métier et aspects technologiques des applications mobiles afin de créer une stratégie de test.
MAT-02 Identifier et comprendre les principaux défis, risques et attentes associés pour le test d'une application mobile.
MAT-03 Appliquer les types de tests et les niveaux spécifiques aux applications mobiles.
MAT-04 Appliquer des types de tests communs, tels que ceux mentionnés dans [ISTQB CTFL 2018], dans le contexte spécifique mobile.
MAT-05 Effectuer les activités requises spécifiquement pour les tests d'applications mobiles dans le cadre des principales activités décrites dans le processus de test ISTQB®.
MAT-06 Identifier et utiliser des environnements et des outils appropriés pour les tests d'applications mobiles.
MAT-07 Comprendre les méthodes et les outils utilisés spécifiquement pour prendre en charge l'automatisation des tests d'applications mobiles.

0.4 Objectifs d'apprentissage examinables

Ces objectifs d'apprentissage soutiennent les objectifs métier et sont utilisés pour créer l'examen qui doit être passé pour obtenir la certification de test d'application mobile au Niveau Fondation.
Les objectifs d'apprentissage sont associés à un niveau cognitif de connaissances (Niveau K). Un niveau K, ou niveau cognitif, est utilisé pour classer les objectifs d'apprentissage en fonction de la taxonomie révisée de Bloom [Anderson 2001]. ISTQB® utilise cette taxonomie pour concevoir les examens de ses syllabi.
Ce syllabus tient compte de trois niveaux K différents (K1 à K3). Pour plus d'informations, voir le chapitre 7.

0.5 Niveaux de compétence pratique

Le Niveau Fondation du syllabus « Test d'application mobile » introduit le concept d'objectifs pratiques qui mettent l'accent sur les compétences pratiques.
Les compétences peuvent être obtenues en effectuant des exercices pratiques, tels que ceux indiqués dans la liste non exhaustive suivante :

- Exercices pour les objectifs d'apprentissage de niveau K3 exécutés à l'aide de papier et de stylo ou d'un logiciel de traitement de texte, comme c'est le cas pour divers syllabi ISTQB® existants.
- Mise en place et utilisation d'environnements de test.
- Tester des applications sur des appareils virtuels et physiques.
- Utilisation d'outils sur les ordinateurs de bureau et/ou les appareils mobiles pour tester ou aider à tester des tâches connexes telles que l'installation, la requête, l'enregistrement, le suivi, la prise de captures d'écran, etc.

Les niveaux suivants s'appliquent aux objectifs pratiques :

- H0 : Cela peut inclure une démo en direct d'un exercice ou d'une vidéo enregistrée. Comme ce n'est pas effectué par le candidat, il ne s'agit pas strictement d'un exercice.
- H1 : Exercice guidé. Les candidats suivent une séquence d'étapes effectuées par le formateur.
- H2 : Exercice accompagné de conseils. Le candidat reçoit un exercice avec des conseils pertinents pour permettre à l'exercice d'être résolu dans les délais donnés.
- H3 : Exercices non guidés, sans conseils.

Recommandations :

- Les objectifs d'apprentissage K1 utilisent généralement le niveau H0 et H1 ou H2 lorsque la situation l'exige.
- Les objectifs d'apprentissage K2 utilisent généralement les niveaux H1 ou H2 et H0 ou H3 lorsque la situation l'exige.

Les objectifs d'apprentissage K3 utilisent généralement les niveaux H2 ou H3, bien qu'il ne soit pas toujours nécessaire d'avoir un exercice pratique pour un objectif d'apprentissage K3. Si la configuration est complexe ou si elle prendra trop de temps, utilisez un niveau H0.

0.6 L'examen

L'examen de certification de test d'application mobile de Niveau Fondation sera basé sur ce syllabus. Les réponses aux questions d'examen peuvent exiger des connaissances fondées sur plus d'une section de ce syllabus. Toutes les sections du syllabus sont examinables, à l'exception de l'introduction et des annexes. Les normes, livres et autres syllabi ISTQB® sont inclus comme références, mais leur contenu n'est pas examinable, au-delà de ce qui est résumé dans ce syllabus lui-même.

L'examen de test d'application mobile de Niveau Fondation est à choix multiple. Il y a 40 questions. Pour réussir l'examen, il faut répondre correctement à au moins 65 % des questions (c.-à-d. 26 questions). Les objectifs et les exercices de compétences pratiques ne seront pas examinés.

Les examens peuvent être passés dans le cadre d'un cours de formation accrédité ou de façon indépendante (p. ex., dans un centre d'examen ou lors d'un examen public). La participation à un cours de formation accrédité n'est pas une condition préalable à l'examen.
Le candidat qui souhaite se présenter à l'examen sans recevoir de formation d'un organisme accrédité devrait lire les lignes directrices sur les compétences fournies dans le document suivant y afférant [CT FL-MAT-2019-Accreditation-and-Competence-Guidelines.pdf] et essayer de mener à bien ces exercices pratiques par lui-même. Cela l'aidera à acquérir les compétences qu'on s'attendrait à ce qu'un organisme de formation accrédité lui prodigue. Veuillez noter que cela n'a aucune incidence sur l'examen de certification de test d'application mobile de Niveau Fondation car ce dernier est exclusivement basé sur ce syllabus et ses objectifs d'apprentissage.

0.7 Temps de formation recommandés

Un temps de formation minimum a été défini pour chaque objectif d'apprentissage dans ce syllabus. Le temps total pour chaque chapitre est indiqué dans la rubrique du chapitre.

Les organismes de formation noteront que les syllabi ISTQB® appliquent une approche de « temps standard » qui alloue un minutage fixe selon le niveau K. Le syllabus test d'application mobile n'applique pas strictement ce système. Par conséquent, les organismes de formation disposent d'une indication plus souple et plus réaliste des temps de formation minimum pour chaque objectif d'apprentissage.

0.8 Conditions d'entrée

La certification ISTQB® de Niveau Fondation doit être obtenue avant de passer cet examen.

0.9 Sources d'information

Les termes utilisés dans le syllabus sont définis dans le glossaire ISTQB® des termes utilisés dans les tests logiciels [ISTQB_GLOSSARY].
Le chapitre 6 contient une liste de livres et d'articles recommandés sur les tests d'applications mobiles.

1. Aspects commerciaux et technologiques du monde Mobile - 175 mins

Mots-clés :
Analyse des risques, atténuation des risques, tests basés sur les risques, stratégie de test

Objectifs d'apprentissage pour les Aspects commerciaux et technologiques du monde Mobile

1.1 Mobiles : Données analytiques.

MAT-1.1.1 (K2) Décrire comment les données d'analyse mobiles disponibles peuvent être utilisées comme entrée pour la stratégie de test et le plan de test.

H0-1.1.1 (H3) Sur la base des données recueillies à partir d'une ou plusieurs sources de données analytiques (localisation géographique, plate-forme, version du système d'exploitation et distribution de type d'appareil), sélectionnez les types d'appareils à tester et leur priorisation correspondante.

Note : H0-I .1 .1 and H0-I .7.1 (ci-dessous) peuvent être combinés.

1.2 Modèles métier pour l'application mobile

MAT-I .2.1 (K2) Distinguer les différents modèles métier pour les applications mobiles.

1.3 Types d'appareils mobiles

MAT-1.3.1 (K 1) Rappeler les différents types d'appareils mobiles.

1.4 Types d'applications mobiles

MAT-1.4.1 (K2) Distinguer les différents types d'applications mobiles.

1.5 Architecture d'application mobile

MAT-1.5.1 (K2) Distinguer les types d'architecture générale des applications mobiles.

1.6 Stratégie de test pour les applications mobiles

MAT-1.6.1 (K3) Appliquer les caractéristiques et les spécificités du marché mobile dans la préparation d'une stratégie de test.

1.7 Défis des tests d'applications mobiles

MAT-1.7.1 (K2) Donnez des exemples des défis associés au test d'applications mobiles.

H0-I .7.1 (H1) Recueillir des données de marché telles que la part de marché d'un appareil ou d'un système d'exploitation pour une région sélectionnée. Recueillir des données pour la taille et la densité de l'écran. Créez une liste de cinq appareils et calculez la couverture du marché prévue pour cette liste.

Note : H0-I .1 .1 (voir ci-dessus) et H0-I .7.1 peuvent être combinés.

1.8 Risques liés aux tests d'applications mobiles

MAT-1.8.1(K2) Décrire comment les risques spécifiques aux applications mobiles peuvent être atténués.

1.1 Données d'analyse mobile

Il existe de nombreuses parties prenantes dans le monde mobile, y compris les fabricants, les fournisseurs de plates-forme, les fournisseurs de systèmes d'exploitation (OS), les fournisseurs de données de marché, les fournisseurs d'outils et, bien sûr, les développeurs d'applications et les testeurs.

Afin de contribuer efficacement aux discussions de planification des tests et à l'analyse des tests, un testeur d'applications mobiles doit être informé et connaître les facteurs suivants :

- Les implications commerciales de la distribution des plateformes.
- Les téléchargements d'applications par plateforme.
- La quantité et la distribution des versions OS.
- La distribution sur le marché de divers types d'appareils, y compris les variations en fonction de l'emplacement géographique.
- Les différentes tailles et résolutions d'écran.
- Les différentes méthodes d'entrée.
- Les types d'appareils photo.

Il existe plusieurs sources d'information pour ce qui précède, à la fois gratuites et commerciales. Il s'agit notamment de StatCounter [URL 1], les fournisseurs d'OS eux-mêmes et d'autres sources tierces.
Les données d'analyse mobile sont utilisées pour sélectionner un portefeuille d'appareils pour l'exécution des tests qui convient au marché cible. Des tests sont effectués sur ce portfolio pour tester l'application sur un appareil en fonction de l'importance de l'appareil. Les données relatives aux appareils et à leurs caractéristiques spéciales, le cas échéant, peuvent également être utilisées pour concevoir des tests spécifiques à un type d'appareil. Par exemple, un appareil avec capteur de battement cardiaque peut nécessiter des cas de test spéciaux.

1 .2 Modèles d'affaire pour les applications mobiles

Il existe plusieurs modèles qui peuvent être utilisés pour monétiser le travail effectué dans la création d'applications mobiles. Il s'agit notamment, de manière non limitative : Freemium, basé sur la publicité, basé sur les transactions, basé sur les frais, et enfin les applications d'entreprise. En outre, des achats intégrés peuvent être appliqués à certains de ces modèles.

Il y a certains avantages et inconvénients pour chacune de ces approches et le testeur doit garder le modèle d'affaires à l'esprit lorsqu'il teste l'application mobile.

Dans un modèle Freemium, les applications sont généralement gratuites, mais les utilisateurs doivent payer pour obtenir des fonctionnalités supplémentaires. Les applications doivent fournir suffisamment de fonctionnalités pour être attrayantes pour les utilisateurs, tout en fournissant des fonctionnalités avancées pour lesquelles un grand nombre d'utilisateurs seraient prêts à payer.
Les applications basées sur la publicité affichent des publicités à l'écran lorsque les utilisateurs interagissent avec celles-ci. Cette stratégie de génération de revenus est plus efficace quand

les applications sont utilisées pendant des périodes relativement longues. Les concepteurs d'interface utilisateur doivent soigner l'affichage des publicités. Elles doivent être suffisamment visibles sans toutefois cacher les parties essentielles de l'application tout en s'assurant que les utilisateurs ne sont pas distraits ou n'aiment pas utiliser l'application.

Les applications basées sur les transactions facturent aux utilisateurs soit un montant par transaction, soit des frais fixes, soit un pourcentage lié à la valeur de la transaction ou [NDT : un mécanisme] similaires. Ce type de modèle d'affaires convient à un nombre limité d'applications seulement et est généralement appliqué pour les applications commerciales et financières telles que les portefeuilles mobiles.

Les applications payantes requièrent aux utilisateurs de payer pour le téléchargement et l'installation de l'application. Il convient de bien évaluer le choix d'un modèle d'affaires payant, car il existe de nombreuses alternatives gratuites ou freemium pour la plupart des types d'applications. La probabilité que les utilisateurs achètent une telle application augmente si elle fournit des fonctionnalités ou une facilité d'utilisation exceptionnelles, ou lorsque des applications concurrentes ne sont pas disponibles.

Les applications gratuites et d'entreprise ne facturent pas leurs utilisateurs. Les applications d'entreprise sont développées pour un usage interne au sein de l'organisation et fournissent une interface aux services fournis. Il existe de nombreuses applications de ce genre disponibles auprès d'organisations telles que les banques ou les entreprises de commerce électronique. Ces applications ne sont généralement pas axées sur la monétisation de l'application elle-même, mais permettent de générer des revenus en dirigeant les utilisateurs vers les services fournis par les organisations.

1 .3 Types d'appareils mobiles

Il existe une variété d'appareils mobiles disponibles qui supporte différents types d'applications.
Les appareils typiques incluent :

- Téléphones de base.
- Téléphones dotés de fonctionnalités.
- Smartphones.
- Tablettes.
- Dispositifs compagnons - y compris les portables et certains appareils IOT (Internet des objets).

Lors des tests, il convient de garder à l'esprit que chaque type d'appareil a des caractéristiques spécifiques pour des besoins particuliers.

Les téléphones de base sont utilisés uniquement pour la téléphonie et les SMS. Ils fournissent très peu d'applications et de jeux intégrés. L'installation d'applications ou la navigation n'est pas possible.

Les téléphones dotés de fonctionnalités [NDT : feature phones] fournissent un support limité pour les applications et l'installation d'applications. Ils fournissent un accès Internet via un navigateur intégré et peuvent avoir du matériel supplémentaire comme un appareil photo.

Les smartphones ajoutent aux téléphones plusieurs capteurs. Le système d'exploitation prend en charge des fonctionnalités telles que l'installation d'applications, le support multimédia et la navigation.

Les tablettes sont similaires aux smartphones mais sont physiquement plus grandes. Elles sont généralement utilisées lorsqu'un écran plus grand est nécessaire ou désiré et peuvent également prendre en charge une durée de vie de la batterie plus longue.

Les appareils compagnons et certains appareils et objets connectés (IOT – Internet of Things) sont des appareils alimentés par ordinateur couramment utilisés avec un smartphone ou une tablette pour étendre les fonctionnalités disponibles ou pour donner accès aux données sur le téléphone ou la tablette d'une manière plus pratique.

Les dispositifs portables [NDT : « wearables »] sont des appareils qui peuvent être portés par les consommateurs. Ceux-ci peuvent servir de compagnons aux appareils existants ou fonctionner indépendamment. Montres et bandes de fitness sont des exemples d'appareils populaires.

1 .4 Types d'applications mobiles

Il existe trois principaux types d'applications mobiles :

- Natif
- Basé sur le navigateur
- Hybride

Chaque type d'application présente certains avantages et inconvénients, ce qui nécessite une décision d'affaires avant de commencer le développement de l'application.

Les applications natives sont développées à l'aide de kits de développement logiciel spécifiques à la plate-forme (SDK – Software Development Kit), d'outils de développement, de capteurs et de fonctionnalités spécifiques à la plate-forme. Elles sont téléchargées, installées et mises à jour à partir des magasins fournisseurs. Ces applications peuvent avoir besoin d'être testées sur tous les appareils pris en charge.

Les applications natives offrent généralement de meilleures performances, peuvent utiliser pleinement les fonctionnalités de la plate-forme et répondre aux attentes de la plate-forme pour laquelle elles sont développées. Le coût de développement est généralement plus élevé et des défis supplémentaires peuvent s'appliquer tels que l'utilisation de plusieurs plates-formes ainsi que l'installation et les tests sur un grand nombre d'appareils.

Les applications basées sur le navigateur sont accessibles via un navigateur mobile. Étant donné que ceux-ci utilisent des technologies de développement web typiques et les navigateurs, le support de plates-formes multiples est facile, et le coût de développement est généralement plus faible.

Il existe quatre façons principales de créer des applications Web mobiles :

- Versions spécifiques mobiles de sites Web et d'applications (ceux-ci sont également connus sous le nom de sites). Habituellement, cela signifie que lorsqu'un navigateur mobile s'adresse à l'application, une version mobile de l'application est fournie. Par exemple, facebook.com redirige vers m.facebook.com lorsqu'il est sollicité à partir d'un appareil mobile.
- Les applications Web réactives [NDT : responsive] garantissent que la conception s'adapte au facteur de forme et à la taille de l'écran, généralement exprimées en tant que fenêtres [NDT : « view port »].
- Les applications Web adaptatives ajustent le design en fonction de certaines tailles prédéfinis. Il existe différents designs pour ces tailles et les fonctionnalités mises à la disposition de l'utilisateur sont souvent réglables.
- Les applications Web progressives permettent de créer des raccourcis de pages Web spécifiques sur l'écran d'accueil mobile. Elles apparaissent comme des applications natives et peuvent parfois même fonctionner hors ligne.

Les applications Web mobiles sont créées à l'aide de technologies Web communes, ce qui les rend généralement plus faciles à développer et à gérer par rapport aux applications natives et hybrides. Elles peuvent toutefois ne pas être aussi riches en fonctionnalités que les applications natives ou hybrides et peuvent avoir un accès limité aux interfaces natives de programmation d'applications (API) de la plate-forme. L'accès aux capteurs mobiles est également limité. Les tests d'installation sur les appareils ne sont pas nécessaires, mais des tests de compatibilité du navigateur sont, en revanche, nécessaires.

Les applications hybrides sont une combinaison d'applications natives et d'applications Web. Elles utilisent un conteneur d'application natif qui contient une fenêtre Web permettant d'exécuter une application Web à l'intérieur d'une application native. Ces applications sont téléchargées à partir des magasins fournisseurs et peuvent accéder à toutes les fonctionnalités de l'appareil. Elles sont relativement faciles à développer, à mettre à jour et à maintenir sans mettre à jour l'application installée sur l'appareil. Les compétences requises pour développer ces applications sont presque les mêmes que pour le développement web. Les possibles points faibles de ces applications incluent des problèmes de performances en raison de l'utilisation d'un conteneur et des divergences possibles par rapport à l'apparence et au ressenti attendus en raison d'aspects spécifiques à la plate-forme.

Les applications natives et hybrides sont installées physiquement sur un appareil et sont donc toujours à la disposition de l'utilisateur, même lorsque l'appareil n'a pas de connexion Internet. En comparaison, les applications basées sur le navigateur nécessitent un accès internet.

Certaines applications sont préinstallées sur l'appareil mobile et d'autres peuvent être installées via divers canaux de distribution, tels que l'App Store d'Apple, google Play Store, les magasins d'applications d'entreprise (disponibles uniquement à l'intérieur du réseau d'entreprise) et les marchés d'applications tierces. Le test de chacun de ces types d'applications peut nécessiter une approche différente. Les paramètres à considérer incluent :

- Les différents types d'appareils à prendre en charge
- Les caractéristiques des capteurs et de l'appareil à utiliser
- La disponibilité dans diverses conditions réseau
- Les caractéristiques de qualité relatives à l'installabilité, la compatibilité, les performances et l'utilisabilité

1.5 Architecture d'application mobile

Il existe de multiples solutions à la conception d'une application mobile.

Certaines des considérations dans le choix d'une architecture ou d'une décision de conception particulière incluent :

- Public cible
- Type d'application
- Prise en charge de diverses plateformes mobiles et non mobiles
- Besoins de connectivité
- Besoins de stockage de données
- Connexions à d'autres appareils, y compris les appareils IOT

Les décisions architecturales comprennent :

- L'architecture côté client comme client léger ou lourd
- L'architecture côté serveur telle qu'une architecture simple ou à multi-tiers
- Le type de connexion tel que Wi-Fi, données cellulaires, communication sans contact (champ proche - NFC), Bluetooth
- Les méthodes de synchronisation des données telles que « store-and-forward », « push » et « pull », communications synchrones et asynchrones.

Les applications client-léger ne contiennent pas de code d'application personnalisé sur l'appareil et utilisent au minimum les fonctionnalités du système d'exploitation mobile. Ces applications utilisent généralement, pour implémenter la logique côté client, le navigateur Web comme frontend et JavaScript comme langage.

Les applications client-lourd peuvent avoir plusieurs couches de code d'application et utiliser des fonctionnalités du système d'exploitation mobile. Il s'agit généralement d'applications natives ou hybrides.

Les architectures côté serveur incluent les possibilités suivantes :

- Les architectures à un seul niveau [NDT : single tier] sont monolithiques et ont tous les serveurs sur la même machine. Ils sont moins évolutifs et plus difficiles à sécuriser.
- Les architectures multi-tiers répartissent les composants côté serveur au travers de différentes unités. Les architectures à deux niveaux impliquent des serveurs Web et de base de données distincts, tandis que les architectures à trois niveaux comprennent également un serveur d'application. Les architectures à plusieurs niveaux permettent la séparation des responsabilités, assurent la spécialisation des bases de données et offrent une meilleure flexibilité, une meilleure évolutivité et une meilleure sécurité. Toutefois, elles peuvent être beaucoup plus coûteuses à développer, gérer et héberger par rapport aux architectures à un seul niveau.

Il existe différentes méthodes de connexion. Un appareil mobile peut être connecté au serveur via des types de connexion tels que le Wi-Fi ou via des connexions de données cellulaires telles que la 2G, la 3G, la 4G et la 5G. Les applications mobiles fonctionnent généralement dans l'un des trois modes suivants :

- Les applications jamais connectées fonctionnent hors connexion et n'ont pas besoin d'être connectées. Une calculatrice simple est un exemple d'une telle application.
- Les applications toujours connectées nécessitent une connexion réseau permanente pendant le fonctionnement. Toutes les applications Web mobiles entrent dans cette catégorie, bien que certaines puissent fonctionner de manière limitée lorsqu'elles sont partiellement connectées.
- Les applications partiellement connectées nécessitent une connexion pour des tâches telles que le transfert de données, mais peuvent fonctionner pendant de longues périodes sans connexion.

La synchronisation des données entre le client et le serveur peut être effectuée dans les modes suivants :

- Le mode continu où les données sont transférées dès qu'elles sont soumises.
- Le mode « store-and-forward » où les données peuvent être stockées localement avant d'être transférées, surtout lorsqu'aucune connectivité n'est disponible.

Le transfert de données peut être effectué dans les deux approches suivantes :

- Le transfert de données synchrones est effectué lorsque la fonction d'appel attend que la fonction appelée soit terminée avant son retour.
- Le transfert de données asynchrone est effectué lorsque la fonction serveur appelée fait un retour immédiat, traite les données en arrière-plan et rappelle la fonction client appelant une fois qu'elle a terminé la tâche. Cela donne aux utilisateurs plus de contrôle. Toutefois, la mise en œuvre du mécanisme de poignée de main augmente la complexité en matière de disponibilité du client ou du réseau lorsque le serveur lance le rappel.

1.6 Stratégie de test pour les applications mobiles

La création d'une stratégie de test pour les appareils mobiles exige que le testeur prenne en compte tous les paramètres énumérés jusqu'à présent dans ce chapitre. De plus, les risques abordés dans cette section et les défis décrits à la section 1.7 doivent également être pris en considération.

Les risques typiques sont, par exemple :

- Sans connaître les données de répartition des appareils dans un emplacement géographique particulier, on ne peut pas choisir les appareils sur lesquels l'application doit être testée de manière durable.
- Sans connaître le type de modèle d'affaires, on ne peut pas tester si le comportement de l'application est bien adapté à ce modèle d'affaires.

La création d'une stratégie de test pour les tests d'applications mobiles doit en outre tenir compte des risques et défis spécifiques suivants :

- La variété des appareils mobiles avec des défauts spécifiques à l'appareil pour certains d'entre eux.
- La disponibilité d'appareils en interne ou via l'utilisation de laboratoires de test externes.
- L'introduction de nouvelles technologies, dispositifs et/ou plates-formes pendant le cycle de vie de l'application.
- L'installation et la mise à niveau de l'application elle-même via différents canaux, y compris la préservation des données et préférences de l'application.
- Les problèmes de plate-forme qui pourraient avoir un impact sur l'application.
- La couverture du réseau et son impact sur l'application dans un contexte mondial.
- La possibilité de tester en utilisant les réseaux de divers fournisseurs de services.
- L'utilisation d'émulateurs mobiles, de simulateurs et/ou d'appareils réels pour des niveaux de test et des types de tests spécifiques.

Ces défis sont décrits plus en détail à la section 1.7.

La stratégie de test prend en compte les risques et les défis. Par exemple :

- La stratégie de test peut spécifier l'utilisation d'émulateurs/simulateurs mobiles dans les premiers stades de développement, suivie par de vrais appareils dans les étapes ultérieures. Il existe certains types de tests qui peuvent être effectués sur les émulateurs mobiles / simulateurs, mais pas tous. Pour en savoir plus à ce sujet, voir la section 4.3.
- La stratégie de test peut tenir compte du défi posé par un grand nombre d'appareils différents en adoptant l'une des approches suivantes :

- Approche de plateforme unique : Réduisez la portée à un seul type d'appareil, à une version OS, à un opérateur et à un type de réseau.
- Approche multiplateforme : Réduire la portée à une sélection représentative d'appareils et de systèmes d'exploitation utilisés par la majorité des clients sur le marché cible, en fonction du trafic mobile ou d'autres données analytiques.
- Approche de couverture maximale : Couvrez toutes les versions, appareils, fabricants, opérateurs, types de réseau et d'OS. Il s'agit essentiellement de tests exhaustifs, qui ne sont généralement pas économiquement viables, surtout lorsque l'on considère la multitude d'appareils et de versions d'OS sur le marché.

- La stratégie de test peut tenir compte du défi posé par la non-disponibilité d'appareils, de réseaux ou de conditions réelles en utilisant des ressources externes, telles que :
  - Services d'accès à distance aux périphériques. Il s'agit d'un moyen d'accéder sur le Web à des périphériques dont on ne dispose pas.
  - Services de test de foule [NDT : crowd testing]. C'est un moyen d'accéder à un grand groupe de volontaires et à leurs appareils.
  - Réseaux personnels tels que des amis et des collègues. Faire usage de son propre réseau social privé.
  - Chasse aux défauts. Il s'agit d'un événement de test gamifié à l'aide de volontaires de l'entreprise ou du grand public.

En plus des niveaux de test décrits dans [ISTQB CTFL 2018], la stratégie de test tient également compte des types courants de tests utilisés pour les applications mobiles (voir section 3.1) et de tous les niveaux supplémentaires de tests requis (voir section 3.2).

1 .7 Défis des tests d'applications mobiles

Dans le monde mobile, de nombreux défis supplémentaires existent qui sont rares ou non critiques dans les logiciels de bureau ou serveur. Les testeurs doivent être conscients de ces défis et de l'impact qu'ils peuvent avoir sur le succès de l'application.

Les défis typiques dans le monde mobile incluent :

- Plates-formes multiples et fragmentation des types d'appareil : plusieurs types et versions d'OS, de tailles d'écran et de qualité d'affichage.
- Différences matérielles dans divers appareils : Divers types de capteurs et difficulté à simuler les conditions de test pour les ressources limitées du processeur et de la RAM.
- Variété d'outils de développement de logiciels requis par les plateformes.
- Différence de conceptions d'interface utilisateur et d'attentes en matière d'expérience utilisateur (UX – User eXperience) par rapport aux plateformes.
- Plusieurs types de réseau et de fournisseurs.

- Appareils disposant de faibles ressources.
- Divers canaux de distribution pour les applications.
- Divers utilisateurs et groupes d'utilisateurs.
- Différents types d'applications avec différentes méthodes de connexion.
- Haute visibilité des retours résultant de défauts ayant un fort impact sur les utilisateurs et qui les poussent facilement à publier des commentaires sur les places de marché en ligne.
- Publication sur les places de marché qui nécessite des cycles d'approbation supplémentaires par les propriétaires de celles-ci tels que Google Play ou Apple App Store.
- Indisponibilité des appareils nouvellement lancés nécessitant l'utilisation d'émulateurs/simulateurs mobiles

L'impact de ces défis inclut :

- Un grand nombre de combinaisons à tester.
- Un grand nombre d'appareils nécessaires pour les tests, ce qui augmente les coûts.
- La nécessité d'une rétrocompatibilité pour exécuter l'application sur les anciennes versions de la plate-forme.
- Nouvelles fonctionnalités publiées dans chaque version du système d'exploitation sous-jacent.
- Lignes directrices à prendre en compte pour les différentes plateformes.
- Processeurs de faibles ressources, quantité limitée de mémoire et d'espace de stockage.
- Bande passante variable et instabilité de divers réseaux.
- Modifications des vitesses de téléchargement et de publication disponibles en fonction des types de tarification.

Les deux exemples suivants illustrent les défis typiques et leur impact potentiel :

- Différents appareils ont différents types de capteurs et les tests doivent en tenir compte. Chaque nouveau capteur ajouté au matériel peut nécessiter des tests de rétrocompatibilité supplémentaires.
- Certains des défis du réseau peuvent être traités de manière appropriée, même dans des conditions réseau variables, en utilisant des stratégies appropriées de mise en cache et de pré-verrouillage [NDT : « pre-fetching »]. Toutefois, cela a un coût : un grand nombre de connexions ouvertes peuvent avoir un impact sur les performances côté serveur car la plupart des applications maintiennent l'utilisateur connecté sur celui-ci.

1 .8 Risques dans le test d'applications mobiles

Les défis mentionnés dans la section 1.7 peuvent apparaître isolément ou en combinaison avec d'autres. Cela peut entraîner des risques supplémentaires pour une application mobile. Un testeur doit être en mesure de contribuer à l'analyse des risques du produit. Les méthodes communes d'analyse et d'atténuation des risques, telles que discutées dans [ISTQB CT FL 2018], chapitre 5.5, peuvent également être appliquées dans le contexte mobile. En outre, les stratégies d'atténuation et de risques spécifiques au mobile suivantes existent :

Risque	Atténuation possible
Fragmentation du marché	Choisissez une sélection appropriée d'appareils pour l'exécution du test, par exemple, tester les appareils les plus couramment utilisés.
Coût de prise en charge de plusieurs plateformes	Effectuer l'analyse pour comprendre la plupart des plates-formes utilisées, évitant ainsi de tester celles qui ne sont plus dans le périmètre
Introduction de nouvelles technologies, plateformes et appareils	Utiliser des versions de préproduction de ces technologies.
Manque de disponibilité des appareils pour l'exécution des tests	Appliquer des services d'accès à distance aux périphériques ou des services de test de foule.
Risques liés aux modèles d'utilisation attendus des applications mobiles utilisées	Appliquer des approches de test appropriées telles que les tests sur le terrain

2. Types de test d'applications mobiles - 265 mins

Mots-clés :

Coexistence, compatibilité, connectivité, compatibilité inter-navigateur, interopérabilité, système sous test (SUT), type de test, utilisabilité

Objectifs d'apprentissage pour les types de tests d'applications mobiles

2.1 Test de compatibilité avec le matériel (des appareils)

MAT-2.1 1	(K2) Décrire les fonctionnalités et le matériel spécifiques à l'appareil qui devraient être pris en compte pour les tests.
HO-2.1.1	(H1) Tester une application pour plusieurs fonctionnalités d'appareils mobiles pendant que le système sous test (SUT) est utilisé afin de vérifier le bon fonctionnement du SUT.
MAT-2.1.2	(K3) Préparer les tests pour la compatibilité de l'application avec la taille de l'écran, le rapport de forme et la densité de l'écran.
HO-2.1.2	(H3) Tester une application sur plusieurs appareils mobiles (virtuels ou physiques) pour montrer l'impact de la résolution et de la taille de l'écran sur l'interface utilisateur de l'application.
MAT-2.1.3	(K2) Décrire comment les tests peuvent montrer les effets potentiels de la surchauffe de l'appareil sur le système sous test.
MAT-2.1.4	(K1) Rappeler différents types de tests pour tester les différents capteurs d'entrée utilisés dans les appareils mobiles.
MAT-2.1.5	(K1) Rappeler les tests à exécuter pour diverses méthodes d'entrée.
HO-2.1.5	(H0) Tester une application pour différents types d'entrées, y compris des tests liés au clavier avec plusieurs claviers installés, des tests gestuels et (en option) des tests liés à la caméra.
MAT-2.1 6	(K2) Décrire comment les tests peuvent révéler les problèmes d'interface utilisateur lors du changement d'orientation de l'écran.
HO-2.1.6	(H3) Tester une application pour vérifier l'effet du changement d'orientation sur les fonctionnalités de l'application, y compris la conservation des données et l'exactitude de l'interface utilisateur.
MAT-2.1.7	(K3) Préparer des tests pour une application à l'aide d'interruptions typiques aux appareils mobiles.
HO-2.1.7	(H3) Tester une application pour plusieurs interruptions d'appareils mobiles pendant l'utilisation de l'application.
MAT-2.1.8	(K3) Préparer des tests modifiant les autorisations d'accès aux fonctionnalités de l'appareil demandées par l'application.
HO-2.1.8	(H3) Tester la gestion des autorisations d'une application en permettant et en refusant les autorisations demandées et en observant le comportement lorsque les dossiers et les paramètres du capteur sont refusés à l'installation ou modifiés après l'installation.
MAT-2.1.9	(K3) Préparer des tests pour vérifier l'impact d'une application sur la consommation d'énergie d'un appareil et l'impact de son état de charge sur l'application.
HO-2.1.9	(H3) Tester une application sous différents niveaux de puissance de la batterie pour découvrir les données de consommation et établir les performances sous des états de batterie faibles et morts.

2.2 Tests des interactions de l'application avec le logiciel de l'appareil

MAT-2.2.1	(K3) Préparez des tests pour le traitement des notifications par le système sous test.
HO-2.2.1	(H2) Tester l'effet de la réception des notifications lorsqu'une application est au premier plan et en arrière-plan. Testez l'effet de la modification des paramètres de notification sur les fonctionnalités de l'application.
MAT-2.2.2	(K2) Décrire comment les tests peuvent vérifier le fonctionnement correct des fonctionnalités des liens à accès rapide.
HO-2.2.2	(H3) Tester les fonctionnalités de raccourci/accès rapide d'une application.
MAT-2.2.3	(K3) Préparer des tests de l'impact sur une application des paramètres de préférence utilisateur fournis par un système d'exploitation.
HO-2.2.3	(H3) Tester une application en cours d'exécution en modifiant les options de valeur d'entrée pour les préférences fournies par le système d'exploitation.
MAT-2.2.4	(K2) Distinguer les différents tests requis pour les applications natives, web et hybrides.
HO-2.2.4	(H0) (Facultatif) Identifier les tests nécessaires pour les applications, selon le type d'application.
MAT-2.2.5	(K1) Rappeler les tests requis pour les applications disponibles sur plusieurs plateformes ou versions de systèmes d'exploitation.
MAT-2.2.6	(K1) Rappeler les tests requis pour la coexistence et l'interopérabilité avec d'autres applications.

2.3 Tests de diverses méthodes de connectivité

MAT-2.3.1	(K2) Résumer les tests à réaliser pour tester la connectivité, y compris ceux entre les réseaux, lors de l'utilisation de Bluetooth et lors du passage au mode avion.
HO-2.3.1	(H0) (Facultatif) Effectuer des tests sur une application qui transfère des données au serveur lorsque le téléphone passe de la connectivité Wi-Fi à la connectivité de données cellulaires en fonction de leurs forces de signal disponibles.

2 .1 Test de compatibilité avec le matériel (des appareils)

2.1.1 Test des fonctionnalités de l'appareil

Différents types d'appareils avec des capacités différentes signifient que les tests de compatibilité doivent être effectués sur un grand nombre d'appareils. Cela nécessite de hiérarchiser les périphériques cibles pour effectuer les tests. Les données du marché servant à la hiérarchisation, telles que discutées dans la section 1.1 sont utilisées pour sélectionner un portefeuille d'appareils le plus approprié pour le marché cible. La sélection de ce portefeuille d'appareils constitue généralement un compromis entre la couverture du marché, le coût et le risque.

Les applications peuvent être installées sur différents types d'appareils avec les fonctionnalités suivantes :

- Différentes méthodes d'arrêt

- Différentes façons de naviguer
- Utilisation de claviers durs et souples
- Diverses fonctionnalités matérielles telles que :
  - Radio
  - USB
  - Bluetooth
  - Appareil photo
  - Haut-parleur
  - Microphone
  - Accès écouteurs

Aucune de ces fonctionnalités ne devrait interférer négativement avec les opérations de l'application.

Les caractéristiques de l'appareil présentent de nombreuses variations qui peuvent différer même entre les différents modèles d'appareils fabriqués par un même fabricant. Elles sont couramment utilisées pour différencier les segments de marché et peuvent changer rapidement au fil du temps. Par exemple, il est actuellement assez courant pour les appareils haut de gamme et milieu de gamme d'avoir des capteurs d'empreintes digitales, tandis que les appareils du bas de gamme n'en ont pas. Cela change avec le temps. Il y a quelques années, les capteurs d'empreintes digitales n'étaient inclus dans aucun appareil mobile. En raison de cette évolutivité, le testeur a besoin d'une compréhension claire des appareils et des fonctionnalités attendues par ses utilisateurs. Le testeur doit créer le portefeuille d'appareils et concevoir les tests correspondants en conséquence.

En général, il ne suffit pas de tester si l'application fonctionne correctement avec les fonctionnalités attendues. Il est nécessaire de tester en plus que l'application fonctionne toujours comme prévu si une certaine fonctionnalité est absente. Par exemple, une application qui prend en charge l'utilisation de la caméra avant et arrière ne doit pas cesser de fonctionner si elle est installée et exécutée sur un appareil disposant de plusieurs caméras, d'une seule caméra ou de pas de caméra du tout.

2.1.2 Tests de différents écrans

Les écrans des appareils peuvent présenter différentes tailles, différentes tailles de fenêtre, rapports de forme et résolutions, mesurés en pixels par pouce (ppi) et en points par pouce (dpi). La grande variété des appareils nécessite une priorisation. Des tests devraient être créés, qui utilisent l'interface utilisateur pour différents appareils, différentes tailles d'écran, de résolutions et de facteurs de forme les plus communs sur le marché cible.

Des tests pour différents écrans doivent être effectués pour vérifier ce qui suit :

- L'application mesure tous les éléments de l'interface utilisateur en fonction de la densité et de la taille actuelles de l'écran.
- Les éléments de l'interface utilisateur ne se chevauchent pas.

- Les problèmes d'utilisabilité ou tactiles ne se produisent pas.
- Il n'y a pas de rétrécissement problématique des images en raison de dpi/ppi élevés.

2.1.3 Tests de la température de l'appareil

Contrairement aux ordinateurs de bureau, les appareils mobiles réagissent différemment à l'augmentation de la température de l'appareil.
Les appareils mobiles pourraient surchauffer pour diverses raisons telles que la charge de la batterie, la charge de travail intense, les applications fonctionnant en arrière-plan, l'utilisation continue des données cellulaires, le Wi-Fi ou le GPS.
La surchauffe peut avoir un impact sur un appareil qui tente de réduire la chaleur pour conserver les niveaux de batterie. Cela peut inclure une baisse de la fréquence du Processeur, la libération de la mémoire, et l'extinction des parties du système.
Si cela se produit, cela peut également avoir un impact sur la fonctionnalité de l'application et doit conséquemment être pris en compte lors de la planification des tests. Les tests doivent être conçus pour consommer beaucoup d'énergie conduisant à la production de chaleur sur une longue période ininterrompue. Le logiciel sous test ne doit alors montrer aucun comportement inattendu.

2.1.4 Tests des capteurs d'entrée de l'appareil

Les appareils mobiles reçoivent une variété de types d'entrées à partir de capteurs qui utilisent, par exemple, le GPS, les accéléromètres, les gyroscopes et les magnétomètres à 3 axes ou qui réagissent à la pression, à la température, à l'humidité, au rythme cardiaque, à la lumière ou aux entrées sans contact.
Les tests pour différents capteurs d'entrée d'un appareil vérifient ce qui suit :

- L'application fonctionne comme prévu pour chacun des capteurs disponibles. Par exemple, l'application doit être testée pour différents types de mouvement tels que le mouvement circulaire et le mouvement d'avant en arrière (comme dans la marche).
- Les caractéristiques qui réagissent à l'éclairage externe réagissent correctement dans diverses conditions d'éclairage.
- Les entrées sonores et les sorties réagissent correctement en conjonction avec des boutons à volume matériels et logiciels, des microphones, des haut-parleurs câblés et sans fil, et dans diverses conditions sonores ambiantes.
- La position de localisation est exacte dans les conditions suivantes :
  - Allumer et éteindre le GPS.
  - Qualité différente du signal GPS.
  - Lorsque l'application a besoin d'un repli sur diverses autres méthodes de détermination de l'emplacement, y compris le Wi-Fi, l'emplacement de la tour cellulaire ou l'entrée manuelle de l'emplacement.

2.1.5 Tests de diverses méthodes d'entrée

Les tests pour différentes méthodes d'entrée des appareils vérifient ce qui suit :

- Étant donné que les téléphones mobiles permettent une variété de claviers logiciels/virtuels à installer, l'application est en mesure de travailler avec au moins ceux fournis par les principaux fabricants d'appareils et ceux qui sont largement utilisés.
- L'application s'assure que le clavier apparaît par défaut avec une mise en page et des touches appropriées au besoin.
- Lorsqu'un utilisateur place un ou plusieurs doigts sur l'écran tactile, l'application interprète ce motif comme un geste ou une commande particulière. Les gestes typiques incluent le presser/toucher, presser à deux doigts, le multi-touch, le balayage/défilement, appuyer une fois [NDT : « tap »], appuyer deux fois [NDT : « double tap »], glisser, pincer zoomer/dézoomer.
- Chaque écran de l'application doit répondre correctement aux gestes ou autres moyens d'entrée appropriés et ignorer tous les gestes ou entrées non pris en charge.
- Les appareils photos utilisées par les applications sont capables de capturer des images et des vidéos, de scanner des codes à barres, des codes QR et des documents, et de mesurer les distances.
- Lorsque des caméras avant et arrière sont disponibles, la caméra appropriée est activée par défaut. Par exemple, lorsqu'un chat vidéo nécessite que la caméra avant soit allumée par défaut, les applications doivent être testées dans les cas où l'application utilise l'entrée de la caméra et dans ceux où elle ne l'utilise pas. En outre, les tests doivent s'assurer que le logiciel testé fonctionne correctement si une seule caméra (avant ou arrière) est présente au lieu de deux. Cela est particulièrement vrai si le logiciel sous test utilise une caméra particulière et que celle-ci manque.

2.1.6 Tests du changement d'orientation de l'écran

Les capteurs de mouvement sont utilisés pour détecter les changements d'orientation et déclencher une bascule entre les modes paysage et portrait (et vice versa) avec des modifications apportées à la disposition de l'interface utilisateur si nécessaire.

Les tests après un changement d'orientation de l'écran doivent vérifier ce qui suit :

- Utilisabilité et comportement fonctionnel corrects lorsqu'un passage au mode portrait ou paysage est effectué.
- L'application maintient son état.
- Les champs de données d'entrée conservent les données déjà saisies.
- Les champs de données de sortie affichent les mêmes données tout en maintenant la session en cours.

Les tests après un changement d'orientation de l'écran ne doivent pas se concentrer uniquement sur un seul changement de mode d'affichage, car les problèmes de rendu ou d'état peuvent ne pas toujours s'afficher après un seul changement. Les tests doivent donc être effectués avec plusieurs bascules ininterrompues entre les modes portrait et paysage.

Des tests qui modifient l'orientation plusieurs fois, dans les différents états d'une interface utilisateur, avec et sans données, doivent être conçus. L'application doit se comporter comme prévu, en maintenant son état sans aucune perte ou changement de données.

2.1.7 Tests des interruptions typiques

Les types courants d'interruptions d'appareil incluent les appels vocaux, les messages, l'allumage du chargeur, la mémoire faible et d'autres notifications. Les interruptions initiées par l'utilisateur résultent d'actions telles que la commutation d'applications ou la mise en veille de l'appareil pendant que l'application est en cours d'exécution

Les tests des interruptions doivent vérifier ce qui suit :

- L'application gère correctement toutes les interruptions mentionnées ci-dessus sans impact négatif sur le comportement de l'application.
- L'application continue de fonctionner correctement, préservant son état, ses données et ses sessions, quelle que soit l'interruption.
- Lorsque l'appareil dispose d'un mode de blocage « ne pas déranger » qui supprime les notifications, l'application doit s'assurer que les différentes conditions sont utilisées correctement. Ces tests doivent également être effectués lorsque le mode « ne pas déranger » est désactivé après avoir été actif pendant une longue période de temps. Il en résulte que de nombreuses notifications sont reçues à la fois.
- Les tests doivent être conçus pour recevoir des interruptions pendant l'utilisation de l'application pour s'assurer que les interruptions n'ont pas d'impact négatif. Par exemple, répondre à un appel téléphonique pendant l'utilisation de l'application et ensuite l'utilisateur revient à l'état où il se trouvait au moment de l'interruption.

2.1.8 Tests des autorisations d'accès aux fonctionnalités de l'appareil

Les applications ont besoin d'accéder à divers dossiers tels que des contacts et des images et à des capteurs tels que l'appareil photo et le microphone. Lorsque l'accès est refusé lors de l'installation ou modifié après l'installation, cela peut avoir un impact sur le comportement de l'application.

Les tests pour les autorisations d'accès vérifient ce qui suit :

- L'application est capable de travailler avec des autorisations réduites ; elle demande à l'utilisateur d'accorder ces autorisations et ne défaille pas de manière inexpliquée.
- Les autorisations ne sont demandées que pour les ressources pertinentes aux fonctionnalités de l'application ; aucune autorisation générale pour les ressources non liées (à son fonctionnement) n'est autorisée.
- La fonctionnalité de l'application répond correctement si une autorisation est retirée ou rejetée lors de l'installation.

- Toute demande d'autorisation émise par l'application est correcte et justifiée.

Pour tester les autorisations d'accès, un testeur doit savoir pourquoi l'application a besoin de chaque autorisation et comment les fonctionnalités doivent être affectées si l'autorisation est retirée ou rejetée lors de l'installation. Le test doit être conçu pour rejeter les autorisations pendant l'installation ainsi que pour accorder des autorisations après l'installation.

2.1.9 Tests de la consommation d'énergie et de l'état

Les tests de consommation d'énergie et de de l'état réalisent la vérification les éléments suivants :

- État de l'alimentation de la batterie et défauts liés à la consommation (de la charge).
- Intégrité des données dans des conditions de faible puissance et de batterie déchargée.
- Consommation d'énergie pendant que l'application est active et est sous une utilisation intensive et faible.
- Consommation d'énergie pendant que l'application est en arrière-plan.

Ces tests doivent être planifiés avec soin car ceux-ci doivent être exécutés sans interruption sur une longue période de temps. Par exemple, l'appareil peut devoir être laissé sans surveillance avec l'application en arrière-plan ou au premier plan alors que l'appareil n'est pas utilisé. Des outils tels que les analyseurs de logs sont nécessaires pour extraire des informations sur les modèles de vidage des batteries.

2 .2 Tests des interactions entre l'application et le logiciel de l'appareil

2.2.1 Tests des notifications

Il existe différents mécanismes utilisés par le système d'exploitation pour afficher les notifications. Parfois, le système d'exploitation retarde l'affichage des notifications ou ne les affiche pas du tout dans le but d'optimiser la consommation d'énergie. Les conditions de test suivantes doivent être prises en considération :

- Le traitement correct des notifications reçues lorsque l'application est au premier plan ou en arrière-plan, en particulier dans de faibles conditions de batterie.
- Si les notifications permettent une interaction directe avec le contenu de l'application (c'est-à-dire sans ouvrir l'application elle-même), l'interaction utilisateur doit être fournie par l'application ultérieurement. Si, par exemple, l'utilisateur répond à une notification, il doit être possible d'accéder à cette réponse à partir de l'application ultérieurement.
- Si les notifications permettent l'accès à l'application, la page correspondante de l'application doit être ouverte et non l'écran d'accueil lorsque la notification contient un lien direct vers cette page.

2.2.2 Tests des liens à accès rapide

Les liens à accès rapide tels que les raccourcis d'application dans Android et Force-touch ou 3d-touch pour iOS peuvent être fournis par le logiciel sous test. Ces fonctionnalités effectuent un sous-ensemble de la fonctionnalité de l'application, à partir de l'écran d'accueil, sans lancer réellement l'ensemble de l'application.

Les conditions de test suivantes doivent être prises en considération :

- Lorsque certaines fonctionnalités ne sont disponibles que sur une version particulière du système d'exploitation, le système sous test doit se comporter correctement s'il est installé sur des versions du système d'exploitation qui offrent ou n'offrent pas de telles fonctionnalités.
- Les actions effectuées dans les liens à accès rapide sont reflétées correctement dans l'application lorsqu'elles sont ouvertes.

2.2.3 Tests des préférences des utilisateurs fournis par le système d'exploitation

Toutes les préférences (paramètres) fournies aux utilisateurs par le système d'exploitation doivent être testées. Les utilisateurs ressentent une expérience négative quand leurs paramètres de préférence ne sont pas respectés par l'application. Par exemple, si l'appareil est mis en sourdine, l'application ne doit pas lire les sons.

Les conditions de test suivantes doivent être prises en considération :

- Les utilisateurs peuvent modifier les options de préférence typiques telles que le son, la luminosité, le réseau, le mode économiseur d'énergie, la date et l'heure, le fuseau horaire, les langues, le type d'accès et les notifications.
- Les applications adhèrent aux préférences définies en se comportant en conséquence.

2.2.4 Tests de différents types d'applications

Des tests spécifiques peuvent être effectués en fonction du type d'application mobile (voir la section 1.4). Les conditions de test suivantes doivent être prises en considération :

- Pour les applications natives :
  - Compatibilité de l'appareil
  - Utilisation des fonctionnalités de l'appareil
- Pour les applications hybrides :
  - Interaction de l'application avec les fonctionnalités natives de l'appareil
  - Problèmes de performances potentiels dus à la couche d'abstraction
  - Utilisabilité (« look and feel ») par rapport aux applications natives sur la plate-forme en question

- Pour les applications Web :
  - Tester pour déterminer la compatibilité inter-navigateur de l'application avec divers navigateurs mobiles communs
  - Vérifier que la fonctionnalité n'est pas affectée en raison de divers moteurs JavaScript
  - Utilisation des fonctionnalités de l'OS (p. ex., sélection de la date et clavier approprié à l'ouverture)
  - Utilisabilité (« look and feel ») par rapport aux applications natives sur la plate-forme en question

2.2.5 Tests d'interopérabilité avec plusieurs plates-formes et versions de systèmes d'exploitation

Les éditeurs de logiciels supportent souvent les applications sur plusieurs systèmes d'exploitation. Chaque système d'exploitation mobile a ses propres limites qui doivent être prises en compte lors des tests d'applications. Les testeurs doivent être conscients des spécificités de chaque plate-forme testée pour s'assurer que l'application fonctionne comme prévu tout en restant conformes à l'apparence et à la sensation de la plate-forme.

Les conditions de test suivantes doivent être prises en considération :

- Traitement des interruptions, notifications et optimisations (p. ex., pour économiser l'énergie).
- Fonctionnalité correcte lorsque les applications multiplateformes partagent du code ou ont été créées à l'aide de frameworks de développement multiplateforme. Notez que si les applications ne partagent pas de code, alors c'est comme tester deux applications différentes et tout doit être testé.
- Test de rétrocompatibilité si une plateforme utilise différentes versions du système d'exploitation.
- Testez les fonctionnalités nouvelles ou modifiées apportées aux plates-formes. Par exemple, dans Android, l'introduction du framework Doze a nécessité des tests sur les différentes versions du système d'exploitation qui supportent ce framework et celles qui ne le font pas.

2.2.6 Tests d'interopérabilité et de coexistence avec d'autres applications sur l'appareil

Il est assez courant pour les applications d'interagir les unes avec les autres lorsqu'elles sont installées sur un appareil. Les exemples typiques sont les applications de contact et de messagerie.

Les conditions de test suivantes doivent être prises en considération :

- Le transfert de données entre le système sous test et l'application utilisée est correct.
- Aucun mal n'est causé aux données de l'utilisateur stockées dans une application utilisée.
- Ananyse de comportements contradictoires. Par exemple, une application peut désactiver le GPS pour économiser de l'énergie, tandis qu'une autre application allume automatiquement le GPS.

Avec des millions d'applications sur le marché, la coexistence ne peut pas être testée de façon réaliste pour tout. Néanmoins, ces problèmes potentiels devraient être pris en considération et testés en fonction de leur risque.

2.3 Tests pour diverses méthodes de connectivité

Les appareils mobiles peuvent utiliser diverses méthodes pour se connecter aux réseaux (voir la section 1.5). Il s'agit notamment des réseaux cellulaires tels que la 2G, la 3G, la 4G et la 5G, ainsi que le Wi-Fi et d'autres types de connexion sans fil tels que NFC ou Bluetooth.

Les alternatives suivantes devraient être envisagées lors de l'exécution de tests de connectivité :

- Les émulateurs/simulateurs d'appareils peuvent simuler diverses connexions réseau et certains fournisseurs de services d'accès à distance aux périphériques les incluent dans leurs fonctionnalités. Les émulateurs/simulateurs ont toutefois une valeur limitée.
- Mettre en place votre propre réseau mobile pour prendre en charge différents types de connexion, puis manipuler la bande passante. Il s'agit d'une alternative très coûteuse.
- Les essais sur le terrain sont potentiellement une alternative plus rentable, mais limités en ce qui concerne la reproductibilité des tests.

En utilisation réelle, les méthodes de connectivité diffèrent. Les utilisateurs peuvent être connectés en permanence à l'aide d'un mode particulier, ou bien ils peuvent passer d'un mode à l'autre, comme du Wi-Fi au cellulaire (par exemple, lorsqu'un utilisateur quitte la maison en utilisant l'application). L'utilisateur peut basculer entre différents réseaux et différentes versions wi-fi/cellulaires, ainsi qu'entre les cellules GSM. Alors qu'ils sont en déplacement, ils peuvent même rencontrer des points morts sans aucun signal réseau. En outre, l'utilisateur peut délibérément se déconnecter en passant, par exemple, en mode avion.

Les tests de connectivité doivent s'assurer que les conditions de test suivantes sont prises en compte :

- Fonctionnalités de l'application correctes avec différents modes de connectivité.
- Le passage d'un mode à l'autre ne provoque aucun comportement inattendu ou perte de données.

- Des informations claires sont données à l'utilisateur si la fonctionnalité est limitée en raison d'une connexion réseau limitée ou non ou si la bande passante est faible. Le message doit indiquer les limites et leurs raisons.

3. Types de tests et processus de test communs pour les applications mobiles — 200 mins

Mots-clés

Fin anormale, accessibilité, injection de code, tests exploratoires, tests sur le terrain, heuristique, installabilité, efficacité des performances, tests de performance, tests après livraison, tests de sécurité, gestion des tests basés sur la session, tests de stress, niveau de test, processus de test, pyramide de test, type d'usage, laboratoire d'utilisabilité, tests d'utilisabilité

Objectifs d'apprentissage pour les types de tests et les processus de test communs pour les applications mobiles

3.1 Types de tests courants applicables aux applications mobiles

MAT-3.1.1	(K3) Préparer des tests d'installabilité pour les applications mobiles.
MAT-3.1.2	(K3) Préparer des tests de stress pour les applications mobiles.
MAT-3.1.3	(K2) Donner des exemples de problèmes de sécurité liés aux applications mobiles.
MAT-3.1.4	(K1) Rappeler les considérations relatives au temps et au comportement des ressources pour les applications mobiles.
MAT-3.1.5	(K3) Préparer des tests d'utilisabilité pour les applications mobiles.
HO-3.1.5	(H2) Choisissez un type d'usage, un mnémonique ou une heuristique pour tester l'utilisabilité d'une application en utilisant la gestion de test basée sur la session. Note : H0-3.1.5 et H0-3.3.1, H0-3.3.2 et H0-3.3.3 peuvent être combinés.
MAT-3.1.6	(K1) Reconnaître le type de tests requis pour tester les bases de données des applications mobiles.
MAT-3.1.7	(K2) Résumer les tests requis pour l'internationalisation (mondialisation) et les tests de localisation des applications mobiles.
MAT-3.1.8	(K2) Résumer la nécessité de tester l'accessibilité dans les tests d'applications mobiles.

3.2 Niveaux de test supplémentaires applicables aux applications mobiles

MAT-3.2.1	(K2) Décrire les niveaux de test supplémentaires, tels que les tests sur le terrain, et les activités supplémentaires associées requises pour des tests d'applications mobiles efficaces.
MAT-3.2.2	(K2) Décrire les tests requis pour effectuer l'approbation du magasin d'applications pour la publication d'applications.

3.3 Techniques de test basées sur l'expérience

MAT-3.3.1	(K 1) Rappeler la gestion des tests basés sur la session, les personae et les mnémoniques dans le contexte des tests mobiles exploratoires.
HO-3.3.1	(H2) Choisissez un mnémonique (ou une partie de celui-ci) qui est spécifique aux tests d'applications mobiles pour tester une application en utilisant la gestion de test basée sur la session. Note : H0-3.1.5 et H0-3.3.1, H0-3.3.2 et H0-3.3.3 peuvent être exécutés ensemble.
MAT-3.3.2	(K2) Décrire l'utilisation des tours et de l'heuristique comme des techniques exploratoires pour les tests d'applications mobiles.
HO-3.3.2	(H2) Choisissez une heuristique spécifique mobile pour tester une application mobile. Note: H0-3.1.5 et H0-3.3.1, H0-3.3.2 et H0-3.3.3 peuvent être exécutés ensemble.
MAT-3.3.3	(K3) Utilisez un type d'usage spécifique au mobile (définissant le type d'usage des fonctionnalités - NDT : « feature tour ») pour tester une application mobile
HO-3.3.3	(H2) Choisissez un type d'usage spécifique au mobile pour tester une application mobile. Note: H0-3.1.5 et H0-3.3.1, H0-3.3.2 et H0-3.3.3 peuvent être exécutés ensemble.

3.4 Processus et approches de test pour le mobile

MAT-3.4.1	(K2) Faire correspondre le processus de test, tel que décrit dans [ISTQB CT FL 2018], aux besoins des tests d'applications mobiles.
MAT-3.4.2	(K2) Décrire les approches de test à chaque niveau de test, spécifiques aux tests d'applications mobiles.

3.1 Types de tests courants applicables aux applications mobiles

3.1 Tests d'installabilité

Les testeurs doivent se concentrer sur l'installation, la mise à jour et la désinstallation de l'application en utilisant les approches suivantes :

- « Application_stores » (magasins d'applications)
  Le processus d'installation peut être différent selon les utilisateurs de l'application. Les utilisateurs peuvent installer l'application à partir de magasins de marché tels que le Google Play Store ou l'App Store d'Apple. Les utilisateurs d'applications d'entreprise devront effectuer des tests d'installation via un lien, ou un service de distribution tel que HockeyApp ou App Center.
- « Sideloading » (copie et installation de l'application)
  Certains systèmes d'exploitation offrent la possibilité d'installer l'application en la copiant sur un appareil mobile et en l'installant à partir du fichier.
- Applications de bureau

Des applications de bureau telles qu'Apple iTunes (pour iOS) ou Android App Installer sont disponibles pour l'installation d'applications sur le smartphone. Le testeur doit y télécharger l'application et utiliser un câble pour l'installer vers le smartphone. La plupart de ces applications de bureau permettent également la désinstallation de l'application.

L'installation peut être effectuée en utilisant les méthodes suivantes :

- OTA (Over-the-Air) via Wi-Fi ou données cellulaires
- Câble de données

Certaines des conditions de test à considérer incluent :

- Installation, désinstallation et mise à niveau de la mémoire interne et externe (si pris en charge).
- Réinstallation de l'application lorsque l'option « conserver les données de l'application » a été choisie lors de la désinstallation précédente.
- Réinstallation de l'application lorsque l'option « conserver les données de l'application » n'a pas été choisie lors de la désinstallation précédente.
- Annuler ou interrompre l'installation ou la désinstallation, par exemple en arrêtant l'appareil mobile pendant le processus ou en se déconnectant d'Internet.
- Reprise de l'installation, de la désinstallation et de la mise à niveau interrompues après l'annulation ou l'interruption.
- Tests liés aux autorisations. Par exemple, certaines applications demandent la permission d'utiliser le carnet d'adresses. Ce test important doit vérifier le comportement de l'application si l'utilisateur refuse l'autorisation. Par exemple, y a-t-il un message correspondant envoyé à l'utilisateur ?
- Mettre à jour l'application et vérifier qu'aucune donnée n'est perdue.

Certaines applications nécessitent des dispositifs « jailbreakés » (iOS) ou « rooted » (Android) qui donnent à l'utilisateur les droits administratifs sur l'appareil. La plupart des fournisseurs de plates-forme ne prennent pas en charge le « jailbreaking / rooting » car ils peuvent entrainer des conséquences juridiques. Une application n'exigeant pas de « jailbreaking / rooting » peut ne pas avoir à être testée pour les dispositifs de « jailbreaking / rooting ».

3.1.2 Tests de Stress

Les tests de stress sont axés sur la détermination de l'efficacité des performances de l'application lorsqu'elle est soumise à des conditions au-delà de la charge normale. Le test de stress dans ce contexte ne s'adresse qu'aux appareils mobiles.
Les tests de stress du backend sont décrits dans le syllabus ISTQB@ tests de performance ([ISTQB CTFL PT 2018]) qui constitue une référence utile pour ce domaine.

Certaines des conditions de test qui peuvent être considérées pour les tests de stress comprennent :

- Utilisation élevée du processeur

- Plus de mémoire disponible
- Faible espace disque
- Stress de la batterie
- Défaillances
- Mauvaise bande passante
- Nombre très élevé d'interactions avec les utilisateurs (les conditions du réseau réel peuvent devoir être simulées pour cela)

Certaines de ces conditions stressantes peuvent être créées à l'aide d'outils tels que Monkey. Il s'agit d'un outil de ligne de commande qui s'exécute sur la ligne de commande shell ADB [URL3] ou, si possible, manuellement, par exemple, en utilisant de gros fichiers ou d'autres applications avec une utilisation élevée du processeur ou la consommation de mémoire.

3.1.3 Tests de sécurité

Étant donné que les tests de sécurité sont un sujet complexe, l'ISTQB a un syllabus Spécialiste distinct à ce sujet [ISTQB CTAL SEC 2016]. Les principaux problèmes de sécurité pour les applications mobiles incluent :

- L'accès aux données sensibles sur l'appareil.
- Le transfert d'informations non crypté ou stockage non sécurisé.

Certaines des conditions de test qui peuvent être considérées pour les tests de sécurité comprennent les sujets suivants :

- Test des entrées pour l'injection de code et le débordement.
- Cryptage des données transférées.
- Cryptage des données stockées localement.
- Suppression de données temporaires après utilisation ou après une fin anormale.
- Effacer le texte dans les champs de mots de passe.

Le top 10 des vulnérabilités liées au mobile de l'Open Web Application Security Project (OWASP) devrait également être exploré [URL2].

3.1.4 Tests de performance

Si l'utilisateur installe l'application et qu'elle n'apparaît pas assez vite (par exemple, en respectant pas un temps inférieur ou égal à 3 secondes), elle peut être désinstallée en faveur d'une autre application alternative. Les aspects de la consommation de temps et de ressources sont des facteurs de succès importants pour une application et les tests de performance sont donc effectués pour mesurer ces aspects.
L'efficacité des performances doit être testée sur l'appareil lui-même en plus de l'interaction avec le système backend et d'autres appareils mobiles.

Les tests de performance de l'ensemble du système doivent être effectués tels que définis dans la stratégie de test et ne sont pas spécifiques au mobile. Veuillez-vous référer au syllabus ISTQB Spécialiste sur les tests de performance [ISTQB_CTFL PT 2018] pour plus de détails.

Le test de performance de l'application elle-même doit contenir la chronométrie pour les flux de travail les plus importants. Voici quelques exemples pour les flux de travail d'une application bancaire en ligne : « Connexion », « Changement d'adresse » ou « Virement bancaire avec PIN et numéro de transaction ». Le testeur doit ensuite comparer cette chronométrie avec des applications similaires.
Outre les mesures chronométriques, il est important de tenir compte des performances perçues par l'utilisateur. L'expérience utilisateur peut avoir un impact énorme sur la durée pendant laquelle l'utilisateur est prêt à attendre qu'une certaine fonction soit terminée.

3.1.5 Tests d'utilisabilité

L'utilisabilité est très importante pour les applications mobiles parce que les données montrent qu'un grand nombre d'utilisateurs désinstallent leurs applications dans les quelques minutes après l'installation en raison d'une mauvaise utilisabilité ou de mauvaises performances, voir [URL4].
Pour cette raison, il est recommandé que la conception de l'expérience utilisateur (UX) considère l'apparence et la sensation [NDT : « look and feel »] de la plate-forme sur laquelle l'application doit être utilisée. Si l'UX n'est pas conforme aux attentes de l'utilisateur pour la plate-forme utilisée, cela peut avoir un fort impact négatif. Ainsi, un testeur doit être conscient de l'apparence et de la sensation sur la plate-forme utilisée.

Les tests d'utilisabilité peuvent être effectués par un testeur à l'aide de diverses heuristiques disponibles et de types d'usage testés.

Considérer l'usage de personae est également pertinent dans le cadre des tests d'utilisabilité. Si nécessaire, un laboratoire d'utilisabilité peut également être utilisé à cette fin.

Dans les projets, les résultats identifiés au cours des tests d'utilisabilité ne sont pour la plupart que des constatations et non des défauts. Le testeur doit avoir la capacité d'expliquer les résultats à l'équipe, au Product Owner ou à des parties prenantes similaires. Pour obtenir une utilisabilité satisfaisante, une application doit :

- Être explicite et intuitive
- Permettre les erreurs de l'utilisateur.
- Être cohérente dans les libellés et le comportement.
- Respecter les directives de conception des plates-formes.
- Rendre les informations nécessaires visibles et accessibles dans chaque taille et type d'écran.

Veuillez-vous référer au syllabus Spécialiste ISTQB Test d'utilisabilité [ISTQB FLUT 2018] pour plus de détails.

3.1.6 Tests de bases de données

De nombreuses applications doivent stocker des données localement à l'aide de divers mécanismes de stockage de données tels que des fichiers plats ou des bases de données.

Certaines des conditions de test à prendre en compte pour le test de base de données des applications mobiles comprennent :

- Validation des problèmes de stockage de données :
  - Synchronisation
  - Conflits de chargement (« upload »)
  - Sécurité des données
  - Contraintes sur les données
  - Fonctionnalité CRUD (Create/Read/Update/Delete) [NDT : créer, lire, modifier, effacer]
  - Recherche
- Test d'intégration des données pour les données fournies par l'appareil (p. ex., contacts) ou par des applications tierces (p. ex., photos, vidéos et messages).
- Performances de stockage des données sur l'appareil.

3.1.7 Tests de l'internationalisation et de la localisation

Les tests d'internationalisation / de mondialisation de l'application incluent le test d'une application pour différents emplacements, formats de dates, de nombres, de devise, et le remplacement des chaînes réelles par des pseudo-chaînes.
Les tests de localisation incluent le test d'une application avec des chaînes localisées, des images et des flux de travail pour une région particulière. Par exemple, les mots russe et allemand peuvent être beaucoup plus longs que ceux d'autres langues. Étant donné que les appareils mobiles ont différentes tailles et résolutions d'écran, des tailles d'écran limitées peuvent entraîner des problèmes avec les chaînes traduites. Ces questions devraient être vérifiées de manière standard lors des tests de mondialisation/localisation.
Un aspect très important à vérifier est le format de date utilisé, comme ANNÉE — MOIS — JOUR ou JOUR - MOIS - ANNÉE.

3.1.8 Tests d'accessibilité

Les tests d'accessibilité sont effectués pour déterminer la facilité avec laquelle les utilisateurs handicapés peuvent utiliser un composant ou un système. Pour les applications mobiles, cela peut être fait en utilisant les paramètres d'accessibilité de l'appareil et en testant l'application pour chaque paramètre.

Les lignes directrices sur l'accessibilité sont disponibles auprès des fournisseurs de plateformes et celles-ci devraient être utilisées. Par exemple, Google [URL5] et Apple [URL6] ont publié des lignes directrices d'accessibilité pour leurs plateformes respectives. Il est également utile de recueillir les commentaires des personnes qui ont besoin d'accessibilité.
Pour le Web mobile, un guide d'accessibilité a été publié par le W3C, qui mérite d'être pris en considération [URL7].

3.2 Niveaux de test supplémentaires applicables aux applications mobiles

En plus des niveaux habituels de tests, des tests unitaires aux tests d'acceptation tels que décrits dans le syllabus ISTQB de niveau Fondation [ISTQB CTFL 2018], il est également nécessaire d'ajouter des niveaux de test pour les tests d'applications mobiles.

3.2.1 Tests sur le terrain

Certaines applications mobiles ont besoin de tests sur le terrain pour s'assurer qu'elles fonctionnent correctement dans le scénario d'utilisation attendu des utilisateurs réels. Cela pourrait inclure des tests sur divers réseaux et sur différents types de technologies de communication telles que le Wi-Fi ou les données cellulaires.

Les tests sur le terrain devraient inclure la prise en compte des types d'usage du mobile, de réseaux, de Wi-Fi et de commutation de données cellulaires pendant l'utilisation de l'application. Les tests doivent être effectués avec des vitesses de téléchargement et des forces de signal variables, et inclure la gestion des angles morts.

Les tests sur le terrain nécessitent une planification minutieuse et l'identification de tous les éléments requis pour les effectuer, tels que les types d'appareils appropriés, le Wi-Fi, les abonnements de données cellulaires pour divers opérateurs et l'accès aux différents modes de transport nécessaires pour assurer une couverture adéquate. En outre, les itinéraires et les modes de transport, ainsi que l'heure de la journée d'exécution des tests, doivent être programmés.

L'utilisabilité d'une application est un autre aspect important qui doit être couvert lors des tests sur le terrain. Ceux-ci devraient intégrer des facteurs environnementaux tels que la température et des conditions similaires liées au scénario d'utilisation.

3.2.2 Tests pour l'approbation du magasin d'applications et tests après livraison (post-release)

Avant qu'une application ne soit envoyée pour publication, certains tests basés sur des checklists doivent être effectués afin d'assurer son approbation par les magasins d'application. Si la version constitue une mise à niveau, alors les tests liés à cette mise à niveau devraient également être exécutés.

Les checklists sont généralement basées sur des guides, telles que celles spécifiques aux systèmes d'exploitation, pour la conception d'interfaces utilisateur, et pour l'utilisation des bibliothèques et des API fournies par les magasins d'applications.
Le processus d'approbation peut prendre un certain temps après la soumission de l'application. Si des problèmes sont trouvés au cours du processus d'approbation, une nouvelle version peut avoir à être soumise, ce qui nécessitera plus de temps. Cette situation nécessite un examen attentif lors de la planification et des tests du projet.
Un autre niveau de test est le test « après livraison ». Les tests à ce niveau incluent le téléchargement et l'installation de l'application à partir des magasins d'applications.

3.3 Techniques de test basées sur l'expérience

3.3.1 Personae et technique mnémonique

Les personae sont des personnages fictifs qui représentent de vrais clients. Ils ont des motivations, des attentes, des problèmes, des habitudes et des objectifs et il est utile de les employer lorsque le comportement réel de l'utilisateur doit être imité.
Une persona peut avoir un nom, un sexe, un âge, un revenu, une formation et un lieu. Dans un contexte mobile, ils peuvent utiliser d'autres applications, vérifier leur appareil mobile x fois par heure et peuvent avoir d'autres appareils et traits personnels.

Un mnémonique est une aide à la mémoire pour se souvenir de quelque chose. Dans le cadre des tests, chaque lettre d'un mnémonique représente une technique, une méthode de test ou un point focal pour les tests. Un exemple d'un mnémonique est SFiDPOT [URL8]. Les lettres dans le mnémonique ont les significations suivantes :

S — Structure (par exemple, les éléments d'interface utilisateur, les autres éléments d'application et leur hiérarchie d'ordre et d'appel)
F — Fonction (par exemple, les caractéristiques souhaitées fonctionnent, sont disponibles et fonctionnent selon les exigences, etc.)
i – Input / Intrant (par exemple, toutes les entrées requises sont disponibles et traitées comme elles devraient l'être, comme les entrées du clavier, des capteurs et de la caméra.)
D — Data (par exemple, les données sont stockées (également sur la carte SD), modifiées, ajoutées et supprimées telles que définies dans les exigences)
P — Plateforme (par exemple, les fonctions spécifiques du système d'exploitation sont disponibles en fonction des paramètres de l'appareil, y compris le magasin pour le téléchargement de l'application)
O — Opérations (par exemple, les activités de l'utilisateur normal sont disponibles, telles que le déplacement entre les réseaux d'opérateurs mobiles et le Wi-Fi)
T — Temps (p. ex., la manipulation et l'affichage des fuseaux horaires, du temps et des dates)

Un mnémonique et heuristique traitant spécifiquement du mobile est I SLICED UP FUN [URL9]. Les lettres dans le mnémonique ont les significations suivantes :

I - Inputs / Intrants
S - Store (magasin)
L - Lieu/Emplacement
I - Interactions et interruptions
C - Communication
E - Ergonomie
D - Data/Donnée
U - Utilisabilité
P - Plateforme
F - Fonction
U - scénarios Utilisateur
N - Network (réseau)

3.3.2 Heuristiques

Une approche heuristique est une approche « empirique » de la résolution de problèmes, de l'apprentissage et de la découverte qui utilise une méthode pratique. Cela ne garantit pas d'être optimal ou parfait, mais peut être considéré comme suffisant pour atteindre les objectifs immédiats.
Il existe de nombreuses heuristiques pour les tests mobiles. La plupart des mnémoniques peuvent être utilisés comme heuristiques, mais toutes les heuristiques ne sont pas des mnémoniques.

3.3.3 Types d'usage (tour)

Les types d'usage (appelé « tour » en anglais) sont utilisés dans les tests exploratoires pour permettre d'explorer une application d'un point de vue et d'une orientation spécifiques. Ils peuvent être effectués pour comprendre le fonctionnement d'une application et pour créer des modèles pour le flux de travail. Les types d'usages (tour) fournissent une méthode efficace pour les tests sur le terrain.

Un exemple de type d'usage est le « tour des points d'intérêt » où un utilisateur imite le parcours dans la ville d'un touriste visitant les monuments bien connus. Le tableau suivant montre comment les visites effectuées lors d'un tour peuvent être utilisées comme analogies pour les étapes à suivre dans les tests mobiles.

Visites du tour des points d'intérêt	Analogie pour les tests mobiles
Le quartier historique	Code patrimonial
Le quartier des affaires Heure de pointe	Logique d'affaire de l'application Démarrage et arrêt de l'application
Le quartier touristique	Une partie de l'application utilisée par les nouveaux arrivants
Le quartier des hôtels	Parties de l'application qui ne sont actives qu'en mode veille

La combinaison de tests basés sur des sessions (voir la section 3.3.4) et de types d'usage, y compris l'utilisation de l'heuristique et de la mnémonique, contribuent à améliorer l'efficacité des tests d'applications mobiles.
Le tableau suivant, pour donner des idées de test possibles, montre quelques bons exemples de tours pour les tests d'applications avec les zones qu'ils couvrent. Certains d'entre eux se trouvent dans [Kohl 17].

Types d'usage (tour) pour les tests d'applications	Sujet couvert
Mannequin	"Look & feel" and utilisabilité

Points d'intérêt	Les fonctionnalités les plus importantes de l'application
Sabotage	Robustesse
Fonctionnel	Nouvelles fonctionnalités
Scénario	Flux de travail entier dans l'application en combinaison avec des User Stories
Connectivité	Connectivité utilisée, telles que Wi-Fi, GSM
Emplacement	Langage correct, dates, nombres
Lumière	Visibilité dans différentes conditions d'éclairage, telles que la lumière rouge foncé, extérieure.
Batterie basse	Pertes de données dans l'application causées par de faibles niveaux d'énergie.
Autres types d'usage pour les tests d'applications	Sujet abordé dans [Kohl 17]
Geste	Utilisez tous les gestes dans la mesure du possible
Orientation	Changer d'orientation
Changez d'avis	Retour
Mouvement	Faire différents types de mouvements
Emplacement	Déplacez-vous
Connectivité	Modifier les types de connexion ou les emplacements en vous déplaçant
Comparaison	Comparer avec d'autres types d'appareils
Cohérence	Vérifiez la cohérence des écrans, GUI

3.3.4 Gestion des tests basés sur la session (« SBTM – session-based test management »)

La gestion des tests basés sur la session permet de gérer les tests exploratoires en blocs de temps. Une session se compose de trois tâches :

- Configuration de session
- Conception et exécution des tests
- Reporting et analyse des défauts

La gestion des tests basés sur la session utilise généralement une feuille de session contenant une charte de test qui fournit des objectifs de test.
En outre, la feuille de session est utilisée pour documenter les activités d'exécution des tests effectuées.

Les tests exploratoires sont une technique de test basée sur l'expérience qui peut être une approche efficace pour tester des applications mobiles. Les techniques de test basées sur l'expérience sont décrites dans [ISTQB CTFL 2018].

3.4 Processus et approches de test mobiles

3.4.1 Processus de test

Les principales activités du processus de test ISTQB sont décrites dans [ISTQB CTFL 2018] et s'appliquent également aux tests d'applications mobiles.

Il y a d'autres aspects spécifiques aux tests mobiles qui devraient toujours être pris en compte dans le cadre du processus de test ISTQB.

Groupe principal d'activités dans le processus de test	Domaines typiques à considérer pour les tests mobiles	Référence dans le syllabus
Planification des tests	• Combinaisons d'appareils qui doivent être testées. • Utilisation d'émulateurs mobiles et de simulateurs mobiles dans le cadre de l'environnement de test. • Défis particuliers dans le test d'applications mobiles. • Types de test spécifiquement requis pour les tests d'applications mobiles	• Section 4.3 • Section 1.7 • Section 3.2
Analyse et conception	• Tests d'approbation dans les magasins d'application • Tests sur le terrain. • Compatibilité de l'appareil. • Type de laboratoires à utiliser. • Types de test spécifiquement requis pour les tests d'applications mobiles	• Section 3.2.2 • Section 3.2.1 • Section 3.2
Mise en œuvre des tests et exécution des tests	• Tests sur le terrain. • Tests d'installation et de téléchargement après livraison. • Techniques basées sur l'expérience.	• Section 3.2.1 • Section 3.1.1 [ISTQB_CTFL_2018]

Mise en œuvre des tests et exécution des tests	• Les tests sont basés sur des lignes directrices des plateformes pour l'interface utilisateur et les magasins d'applications. • Les tests basés sur les lignes directrices seront généralement exécutés par les fournisseurs de plate-forme au cours de leur processus d'approbation du magasin d'applications. • Il est recommandé de les exécuter en tant que fournisseur d'applications avant de les remettre aux fournisseurs de plate-forme afin d'éviter un possible rejet.	

3.4.2 Approches de test

Les tests d'applications mobiles comprennent les activités à effectuer par les développeurs ainsi que par les testeurs.

Il est important de déterminer la profondeur appropriée des tests par niveau de test (c.-à-d. test unitaires, test d'intégration, test système, test sur le terrain, approbation du magasin d'applications, tests post-version et acceptation par l'utilisateur) pour la livraison de produits de bonne qualité. La profondeur des tests nécessaires par niveau de test dépend de nombreux facteurs, tels que l'architecture de l'application, la complexité de l'application et l'audience utilisateur prévue.

Les plateformes de développement mobile fournissent une variété d'outils pour faciliter les tests aux différents niveaux.

Il est très important de comprendre les outils et la façon dont ils peuvent être appliqués à un niveau donné. Par exemple, un simulateur mobile et/ou un émulateur mobile peuvent être utilisés au niveau des tests unitaires s'il est nécessaire de tirer parti des API du framework ou d'instrumentation fournies par la plate-forme. En outre, les simulateurs mobiles et/ou les émulateurs mobiles peuvent être utilisés au niveau des tests système lorsque les appareils réels ne sont pas disponibles. Cela permet de tester les fonctionnalités, des aspects limités de l'utilisabilité ainsi que l'interface utilisateur.

En outre, la mise en œuvre rapide peut servir de point clé pour s'assurer que les périphériques sont configurés correctement et que toutes les conditions préalables à l'exécution seront remplies à temps.
Les tests unitaires et d'intégration sont également importants, ainsi que les tests manuels (en particulier dans l'étape des tests sur le terrain). Il est très courant pour les applications mobiles de renverser la pyramide de test [Knottl 5]. Cela signifie qu'il peut y avoir de nombreux tests manuels.

4. Plateformes d'applications mobiles, outils et environnement — 80 mins

Mots-clés :
Émulateur, tests sur le terrain, tests basés sur la proximité, laboratoire de tests à distance, simulateur

Objectifs d'apprentissage pour les plateformes d'applications mobiles, outils et environnement

4.1 Plateformes de développement pour applications mobiles

MAT-4.1.1 (K1) Rappeler les environnements de développement utilisés pour le développement d'applications mobiles.

4.2 Outils communs de plate-forme de développement

MAT-4.2.1 (K 1) Rappeler certains des outils communément fournis dans le cadre des plateformes de développement d'applications.

H 0-4.2. I (H1) Utilisez des outils du kit de développement logiciel pour prendre des captures d'écran, extraire un journal et simuler des événements entrants.

4.3 Émulateurs et simulateurs

MAT-4.3.1 (K2) Comprendre les différences entre émulateurs et simulateurs

MAT-4.3.2 (K2) Décrire l'utilisation d'émulateurs et de simulateurs pour les tests d'applications mobiles.

H0-4.3.2 (H1) Créez et exécutez un appareil simulé/émulé, installez une application et exécutez quelques tests dessus.

4.4 Mise en place d'un laboratoire de test

MAT-4.4.1 (K2) Distinguer les différentes approches de la mise en place d'un laboratoire de test.

4.1 Plateformes de développement pour applications mobiles

Des environnements de développement intégrés (IDE - Integrated development environments) sont disponibles sur le marché pour divers développements d'applications mobiles. Ces IDE disposent de divers outils qui aident à concevoir, coder, compiler, installer, désinstaller, surveiller, émuler, enregistrer et tester des applications.

Par exemple, Android Studio peut être utilisé pour le développement d'applications Android alors que pour le développement d'applications iOS, Xcode peut être utilisé. Ceux-ci diffèrent des IDE normaux par le soutien supplémentaire qu'ils offrent aux les plates-formes mobiles.

Certains frameworks de développement inter-plateformes sont également disponibles, ce qui contribue au développement d'applications mobiles. Ceux-ci s'exécutent sur plusieurs plates-formes et ne nécessitent pas spécifiquement de codage.

4.2 Outils communs de plate-forme de développement

Les kits de développement de logiciels fournissent généralement divers utilitaires qui sont utiles dans le développement et le test d'applications. Ces utilitaires couvrent un large éventail de fonctionnalités, telles que la prise de captures d'écran, l'extraction de journaux, l'envoi d'événements aléatoires et de notifications à l'appareil, la surveillance de divers paramètres tels que la mémoire et l'utilisation du processeur, et la création d'appareils virtuels.

Des exemples de ces outils sont Android Virtual Device (AVD) Manager, Android Debug Bridge (ADB) ou Android Device Monitor pour Android et Instruments pour iOS.

4.3 Émulateurs et simulateurs

4.3.1 Vue d'ensemble des émulateurs et simulateurs

Dans le cadre de ce syllabus, les termes émulateur et simulateur se réfèrent à l'émulateur mobile ou au simulateur mobile. Les termes simulateur et émulateur sont parfois erronément utilisés de façon interchangeable. Pour les définitions, veuillez consulter le glossaire au chapitre 8.
Un simulateur modélise l'environnement d'exécution [NDT : runtime], tandis qu'un émulateur modélise le matériel et utilise le même environnement d'exécution que le matériel physique. Les applications testées sur un simulateur sont compilées dans une version dédiée, qui fonctionne sur le simulateur mais pas sur un appareil réel. Ainsi, il est indépendant du système d'exploitation réel.
En revanche, les applications compilées pour être déployées et testées sur un émulateur sont compilées avec le « byte-code » réel qui pourrait également être utilisé par le périphérique réel.
Les simulateurs et émulateurs sont très utiles au stade précoce du développement, car ceux-ci s'intègrent généralement aux environnements de développement et permettent un déploiement, des tests et une surveillance rapide des applications.

Les simulateurs sont parfois également utilisés en remplacement de vrais appareils lors des tests. Toutefois, cela reste plus limité que l'utilisation d'émulateurs, car l'application testée sur un simulateur diffère par son « byte-code » de l'application qui sera distribuée.

Les émulateurs sont également utilisés pour réduire le coût des environnements de test en remplaçant des appareils réels pour certains des tests. Un émulateur ne peut pas remplacer complètement un appareil parce que l'émulateur peut se comporter d'une manière différente de l'appareil mobile qu'il essaie d'imiter. En outre, certaines fonctionnalités peuvent ne pas être prises en charge telles que « (multi)touch », accéléromètre, et d'autres. Ceci est en partie causé par les limitations de la plate-forme utilisée pour exécuter l'émulateur.

4.3.2 Utilisation d'émulateurs et de simulateurs

L'utilisation d'émulateurs et de simulateurs pour les tests mobiles peut être utile pour diverses raisons.
Chaque environnement de développement de système d'exploitation mobile est généralement livré dans un pack qui comprend ses propres émulateur et simulateur. Des émulateurs et simulateurs tiers sont également disponibles.

Un testeur peut utiliser n'importe quel émulateur ou simulateur qui convient à son objectif. L'utilisation de l'émulateur ou du simulateur nécessite de les lancer, d'y installer l'application nécessaire, puis de tester l'application comme si elle se trouvait sur l'appareil réel.

Habituellement, les émulateurs et les simulateurs permettent le réglage de divers paramètres d'utilisation. Ces paramètres peuvent inclure l'émulation réseau à différentes vitesses, les forces du signal et les pertes de paquets, la modification de l'orientation, la génération d'interruptions et les données de localisation GPS. Certains de ces paramètres peuvent être très utiles parce qu'ils peuvent être difficiles ou coûteux à reproduire avec des appareils réels, comme des positions GPS globales ou la force du signal.

Se connecter aux émulateurs à des fins d'installation peut nécessiter l'utilisation d'outils en ligne de commande tels que pour Android Debug Bridge (ADB), pour Android, ou de se connecter depuis l'environnement de développement intégré comme avec Xcode ou Android Studio.

4.4 Mise en place d'un laboratoire de test

Les approches suivantes sont utilisées pour mettre en place un laboratoire de test mobile.

Laboratoire sur site
Avec un laboratoire sur site, tous les appareils, émulateurs et simulateurs sont situés sur place. La sélection de l'appareil peut se faire sur la base de divers facteurs tels que le rang de diffusion de l'appareil (comme on le trouve dans Google ou d'autres analyses), le système d'exploitation et les versions, les dimensions et la densité de l'écran, la disponibilité et le coût, les caractéristiques spéciales et l'importance dans le public ciblé.
Les avantages du laboratoire sur site comprennent la disponibilité des appareils pour des tests spécifiques basés sur la proximité et pour des aspects spécifiques aux capteurs tels que la batterie, le toucher ainsi qu'une sécurité accrue.
La mise en place de ce type de laboratoire peut nécessiter des budgets importants en fonction des appareils à se procurer et à entretenir. D'autres défis incluent la disponibilité au moment opportun et les difficultés liées aux tests dans différents endroits et environnements.

Laboratoire de test à distance
Ces laboratoires sont importants et utiles pour les tests lorsque les appareils ou les réseaux ne sont pas physiquement disponibles sur place. L'accès à distance aux périphériques permet l'accès, via une connexion réseau, à divers appareils hébergés dans le centre de données du

fournisseur. Chaque fournisseur potentiel doit être évalué en fonction de la conformité aux exigences, en particulier pour la sécurité.
Certains laboratoires distants fournissent les fonctionnalités supplémentaires suivantes :

- Versions dédiées d'appareils physiques (p. ex., laboratoire d'appareils mobiles Samsung).
- Dispositifs génériques pour un système d'exploitation particulier et une version particulière seulement.
- Bras robotiques pour effectuer des opérations liées au toucher et au geste.
- Connexions réseau privé virtuel (VPN) pour donner accès à l'appareil.
- Connexions cellulaires avec divers opérateurs de réseaux cellulaires.
- Outils et services d'automatisation.

Certains des facteurs à garder à l'esprit lors de l'utilisation de laboratoires de test à distance comprennent la faible réactivité de l'appareil et les options limitées pour interagir avec des appareils tels que le multi-toucher et les gestes. Cette solution peut être rentable pour une utilisation sporadique, mais est généralement plus coûteuse si elle est utilisée pendant de longues périodes de temps pour un large éventail d'appareils.

D'autres facteurs incluent la disponibilité, à la demande, de la plate-forme eu égard à la nécessité d'obtenir l'accès aux appareils manquants dans le laboratoire local et l'évolutivité du laboratoire, qui peut croître et diminuer à mesure que le projet évolue.

Les scénarios de test qui incluent des capteurs tels que NFC/Bluetooth ou la consommation de batterie sont souvent difficiles à tester dans le cloud. Toutefois, différents emplacements géographiques pour les laboratoires distants peuvent faciliter les tests qui nécessitent des connexions réseau et GPS.

Un laboratoire de test peut utiliser l'une ou l'autre combinaison des deux approches, selon les types de tests qui doivent être effectués.

5. Automatisation de l'exécution des tests - 55 mins

Mots-clés :

test basé sur l'appareil, rapport de test, test basé sur l'utilisateur-agent

Objectifs d'apprentissage pour l'automatisation de l'exécution des tests

5.1 Approches d'automatisation

MAT-5.1.1 (K2) Faire la distinction entre les approches d'automatisation courantes et les frameworks pour les tests d'applications mobiles.

5.2 Méthodes d'automatisation

MAT-5.2.1 (K2) Décrire diverses méthodes d'automatisation pour tester les applications mobiles.

5.3 Évaluation des outils d'automatisation

MAT-5.3.1 (K1) Rappeler les différents paramètres à prendre en compte lors de l'évaluation des outils d'automatisation des tests mobiles.

5.4 Approches pour la mise en place d'un laboratoire de test d'automatisation

MAT-5.4.1 (K2) Faire la distinction entre les approches courantes de création de laboratoires de test avec les avantages et les inconvénients en ce qui concerne l'automatisation des tests.

5.1 Approches d'automatisation

Il existe diverses approches et frameworks d'automatisation qui peuvent être utilisés dans les tests d'applications mobiles. Le choix de l'approche sera en partie déterminé par le type d'application.
Deux approches courantes d'automatisation des tests utilisées sont :

- Tests basés sur l'agent utilisateur
- Tests basés sur l'appareil

Les tests basés sur l'agent utilisateur utilisent la chaîne d'identification utilisateur-agent envoyée par le navigateur pour usurper un navigateur particulier sur un appareil particulier. Cette approche peut être utilisée pour l'exécution d'applications Web mobiles. Les tests basés sur l'appareil, d'autre part, impliquent l'exécution de l'application sous test directement sur l'appareil. Cette approche peut être utilisée pour tous les types d'applications mobiles.

Le type d'application peut également déterminer le cadre qui convient à l'automatisation des tests de cette application. Le Web mobile peut être testé à l'aide des outils habituels d'automatisation des applications Web sur le bureau, tandis que les applications natives

peuvent nécessiter des outils spécifiques. Les fournisseurs de plateformes peuvent également fournir des outils d'automatisation dédiés à la plateforme.

Les approches d'automatisation utilisées pour les applications conventionnelles s'appliquent souvent également aux applications mobiles. Il s'agit notamment de capture / playback, pilotées par les données, pilotées par les mots clés et des tests basés sur le comportement tel que décrit dans le syllabus ISTQB® de Niveau Fondation [ISTQB CTFL 2018] et dans le syllabus ISTQB® Spécialiste Automatisation de test Ingénierie [ISTQB CTAL TAE 2016].

Les principales fonctionnalités qu'un framework de test d'applications mobiles devrait généralement inclure sont :

- Identification d'objets
- Opérations sur les objets
- Rapports de test
- Interfaces de programmation d'applications et fonctionnalités extensibles
- Documentation adéquate
- Intégrations avec d'autres outils
- Indépendance avec la pratique du développement de tests

5.2 Méthodes d'automatisation

Pour développer des tests automatisés, le testeur doit comprendre le mécanisme d'enregistrement ou de création de script d'automatisation, et comment accéder et interagir avec les objets graphiques de l'application tels que les boutons, les listes [NDT : listboxes] et les champs d'entrée.

Plusieurs méthodes existent pour identifier un objet graphique utilisé pour l'automatisation des tests mobiles. Il s'agit notamment de la reconnaissance d'image, de la reconnaissance OCR/texte et de la reconnaissance d'objets (web ou native, selon le type d'application).
Un testeur d'applications mobiles doit non seulement pratiquer la détection et l'identification d'objets graphiques, mais aussi comprendre quelle méthode d'identification d'objets sera la plus à même de permettre l'exécution de tests réussis sur une grande variété d'appareils mobiles, en parallèle et en continu.

Les principales différences entre les méthodes de création de scripts sont :

Point de comparaison	Identification d'objets	Comparaison image/OCR
Fiabilité	Tant que l'identificateur est constant, la disposition de l'écran peut être modifiée. Le risque est que des objets peuvent être identifiés dans le code et permettre une interaction tout en étant cachés	Les images peuvent être mises à l'échelle en fonction de la taille de l'écran, mais les tests échouent dès que la mise en page change.

	pour l'utilisateur. Cela peut conduire à des résultats de tests comprenant de faux négatifs.	
Expérience utilisateur	Habituellement, le test manuel est nécessaire, au moins pour améliorer les scripts enregistrés à des fins de lisibilité et de maintenabilité.	Tests complets basés sur l'interface graphique sans avoir besoin de script.
Vitesse d'exécution	Tend à être plus rapide que la comparaison Image / OCR, en particulier lors de l'utilisation d'outils natifs fournis par le fabricant du système.	Tend à être plus lent en raison de la nécessité de comparer l'écran avec une image de base, pixel par pixel.
Maintenance	Dépend de la qualité des scripts de test.	Principalement en fournissant des images de base modifiées.
Défis d'édition	Connaissance requise du langage de script et des méthodes de conception logicielle pour construire une solution d'automatisation durable.	Génération d'images de base, en particulier lorsque l'application change souvent.

5.3 Évaluation des outils d'automatisation

Pour réussir à créer des solutions d'automatisation des tests, les équipes d'automatisation des tests doivent choisir un ensemble approprié d'outils. Il faut prendre en considération les principales différences entre les outils disponibles et leur adéquation aux exigences du projet. (Voir aussi [ISTQB CTAL TAE 2016]).

Les paramètres d'évaluation des outils d'automatisation des tests peuvent être divisés en deux catégories :

- Adéquation à l'organisation
- Adéquation à la technique

Les paramètres d'ajustement organisationnel sont décrits au chapitre 6.2 du syllabus ISTQB® de Niveau Fondation [ISTQB_CTFL 2018].
Les paramètres d'ajustement technique incluent les paramètres suivants :

- Exigences de test d'automatisation sur mobile et les complexités telles que l'utilisation de nouvelles fonctionnalités comme FaceID, les empreintes digitales et les « chatbots » par l'application.

- Exigences de l'environnement de test, telles que les différentes conditions réseau, l'importation ou la création de données de test et la virtualisation côté serveur.
- Rapports de test et capacités de boucle de rétroaction.
- La capacité du framework à gérer et conduire l'exécution à grande échelle, soit localement, soit dans un laboratoire de test dans le cloud.
- Intégration du framework de test avec d'autres outils utilisés dans l'organisation.
- Disponibilité du support et de la documentation pour les mises à niveau actuelles et futures.

5.4 Approches pour la mise en place d'un laboratoire de test d'automatisation

Lors de l'exécution des tests d'applications mobiles, les développeurs et les testeurs ont le choix du laboratoire de test des appareils qu'ils utiliseront pour cibler leur automatisation des tests :

Laboratoire de test d'appareils sur site
Laboratoire de test d'appareils distant

Diverses combinaisons de ces approches peuvent être appliquées. Leurs principales caractéristiques sont décrites et comparées dans la section 4.4.

Les laboratoires de test d'appareils sur site sont généralement plus difficiles et plus chronophages à entretenir. Disposer localement des appareils, en parallèle avec des émulateurs et des simulateurs, est surtout utile lors des phases de développement et de test précoces de l'application mobile.

Lorsqu'elles atteignent un stade plus avancé du développement de l'application, les équipes doivent effectuer des tests de régression complets, des tests fonctionnels et des tests non fonctionnels. Ces tests sont mieux exécutés dans un laboratoire comprenant l'ensemble des appareils. C'est à ce titre qu'un laboratoire de test d'appareils distant est géré, continuellement mis à jour et maintenu dans le cloud. Ces laboratoires de test d'appareils distants complètent un laboratoire de test d'appareils sur site et veillent à ce que des combinaisons suffisantes d'appareils et de systèmes d'exploitation soient disponibles et à jour. En profitant des laboratoires de test d'appareils distants couramment disponibles, les équipes ont accès à un plus grand ensemble de fonctionnalités prises en charge, des rapports de test plus riches et des capacités avancées d'automatisation des tests.

Enfin, lors de l'exécution à l'échelle par le biais d'un framework d'automatisation des tests ou par le biais d'un job d'intégration continue (CI), la stabilité du laboratoire de test dans son ensemble est essentielle à l'efficacité et à la fiabilité des tests. Ces laboratoires sont généralement conçus pour s'assurer que les appareils et les systèmes d'exploitation sont toujours disponibles et stables.

Les laboratoires de test d'appareils distants ne sont pas toujours nécessaires dans les phases de développement ultérieures de l'application. Les laboratoires de test d'appareils sur site,

bien conçus et entretenus, peuvent être aussi bons voire meilleurs que n’importe quel laboratoire de test d’appareils distant.

6. Références

6.1 Documents ISTQB®

[ISTQB_CTFL_2018]:
Testeur Certifié ISTQB® — Syllabus Niveau Fondation — Version 2018
[ISTQB_FLAT_2014]:Testeur Certifié ISTQB® — Syllabus Niveau Fondation — Testeur agile — Version 2014
[ISTQB_FLUT_2018]:
Testeur Certifié ISTQB® — Syllabus Spécialiste — Test d'utilisabilité — Version 2018
[ISTQB CTFL PT_2018]:
Testeur Certifié ISTQB® — Syllabus Spécialiste — Test de performance — Version 2018
[ISTQB CTAL SEC 2016]:
ISTQB@ Certified Tester — Advanced Level Specialist Syllabus — Security Testing – Version 2016
[ISTQB CTAL TAE 2016]:
Testeur Certifié ISTQB® — Syllabus Spécialiste — Automatisation de Test - Ingénierie — Version 2016
[ISTQB GLOSSARY] :
ISTQB® Glossaire des termes utilisés dans les tests logiciels, Version 3.2

6.2 Ouvrages de référence

- [Knottl 5] Knott, D., "Hands-On Mobile App Testing", Addison-Wesley Professional, 2015, ISBN 978-3-86490-379-3
- [Kohl17] Kohl, J. , "Tap into mobile application testing", leanpub.com, 2017, ISBN 978-0-9959823-2-1

6.3 Autres livres et articles

- Boris Beizer, "Black-box Testing", John Wiley & Sons, 1995, ISBN 0-471-12094-4
- Rex Black, "Agile Testing Foundations", BCS Learning & Development Ltd: Swindon UK, 2017, ISBN 978-1-78017-33-68
- Rex Black, "Managing the Testing Process"(3e), John Wiley & Sons: New York NY, 2009, ISBN 978-0-470-40415-7
- Hans Buwalda, "Integrated Test Design and Automation", Addison-Wesley Longman, 2001, ISBN 0-201-73725-6
- Lee Copeland, "A Practitioner's Guide to Software Test Design", Artech House, 2003, ISBN 1-58053-791-X
- Rick David Craig, Stefan P. Jaskiel, "Systematic Software Testing", Artech House, 2002, ISBN 1-580-53508-9

6.4 Liens (Web/Internet)

Avertissement : Tous les liens étaient actifs au 5 Janvier 2019

- [URL 1] http://gs.statcounter.com/

- [URL2] www.owasp.org
- [URL3] https://developer.android.com/studio/test/monkey
- [URL4] https://techcrunch.com/2016/05/31/nearly-1-in-4-people-abandon-mobile-apps-after-only-one-use/
- [URL5] https://www.google.com/accessibility/
- [URL6] https://www.apple.com/uk/accessibility/
- [URL7] https://www.w3.org/WAI/standards-guidelines/mobile/
- [URL8] https://www.slideshare.net/karennjohnson/kn-iohnson-2012-heuristics-mnemonics
- [URL9] http://www.k0hl.ca/articles/ISLICEDUPFUN.pdf

7. Appendice A — Objectifs d'apprentissage/niveau cognitif des connaissances

Les objectifs d'apprentissage suivants sont définis comme s'appliquant à ce syllabus. Chaque sujet du syllabus sera examiné en fonction de l'objectif d'apprentissage.

7.1 Niveau 1 : Se souvenir (K1)

Le candidat reconnaîtra, se souviendra et rappellera un terme ou un concept.
Mots-clés : Identifier, se souvenir, récupérer, rappeler, reconnaître, savoir

Exemples :

Peut reconnaître la définition de « défaillance » comme :

- « Non-prestation de services à un utilisateur final ou à tout autre intervenant » ou
- « Déviation du composant ou du système par rapport à sa prestation, service ou résultat prévu »

7.2 Niveau 2 : Comprendre (K2)

Le candidat peut sélectionner les raisons ou explications des énoncés liés au sujet, et peut résumer, comparer, classer, catégoriser et donner des exemples pour le concept de test.

Mots-clés : Résumer, généraliser, réduire, classer, comparer, cartographier, contraster, illustrer, interpréter, traduire, représenter, déduire, conclure, catégoriser, construire des modèles

Exemples :

Peut expliquer pourquoi l'analyse et la conception des tests devraient avoir lieu le plus tôt possible :

- Pour trouver des défauts quand ils sont moins chers à corriger
- Pour trouver d'abord les défauts les plus importants

Peut expliquer les similitudes et les différences entre le test d'intégration et le test système :

- Similitudes : les objets de test pour les tests d'intégration et les tests système comprennent plus d'un composant, et les tests d'intégration et les tests système peuvent inclure des types de tests non fonctionnels
- Différences : les tests d'intégration se concentrent sur les interfaces et les interactions, et les tests système se concentrent sur les aspects de l'ensemble du système, tels que le traitement de bout en bout

7.3 Niveau 3 : Appliquer (1<3)

Le candidat peut sélectionner l'application correcte d'un concept ou d'une technique et l'appliquer à un contexte donné.

Mots-clés : Implémenter, exécuter, utiliser, suivre une procédure, appliquer une procédure
Exemples :

- Peut identifier les valeurs limites pour les partitions valides et non valides
- Peut sélectionner des cas de test à partir d'un diagramme de transition d'état donné afin de couvrir toutes les transitions

Référence : (for the cognitive levels of learning objectives) : Anderson, L. W. and Krathwohl, D. R. (eds) (2001) A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives, Allyn& Bacon: Boston MA

8. Appendice B — Glossaire des termes spécifiques au domaine

Terme du glossaire	Définition
2G	2ème génération de technologie mobile de télécommunications sans fil.
3d-touch	Voir « Force Touch »
3G	3ème génération de technologie mobile de télécommunications sans fil
4G	4ème génération de technologie mobile de télécommunications sans fil
5G	5ème génération de technologie mobile de télécommunications sans fil.
ADB	Android Debug Bridge (ADB) - outil de ligne de commande qui permet la communication avec un appareil.
Applications basées sur la publicité	Un modèle de monétisation des applications où les organisations de développement gagnent de l'argent par des publicités affichées dans l'application.
Android Device Monitor (ADM)	Un outil autonome qui fournit une interface utilisateur pour les outils de débogage et d'analyse des applications Android.
Android Studio	L'environnement officiel intégré des développeurs (IDE) pour Android. Android Studio fournit des outils pour construire des applications sur tous les types d'appareils Android.
Raccourci de l'application	Un raccourci vers un ensemble spécifique d'actions définies dans une application par les développeurs d'application sur Android 7.1 ou plus.
Magasin d'applications	Une plate-forme de distribution d'applications où les développeurs peuvent télécharger leurs applications et les utilisateurs peuvent rechercher des applications à télécharger et à installer sur leur plate-forme.
Rapport d'aspect	Le rapport largeur/hauteur d'un écran ou d'une image.
Communication asynchrone	Un type de communication dans lequel les données peuvent être transmises de façon intermittente plutôt que dans un flux stable.
AVD	Acronyme pour Android Virtual Device.
Système backend	Un serveur qui fournit des fonctionnalités pour d'autres systèmes.
Application d'arrière-plan	Une application qui est en cours d'exécution en arrière-plan.
Rétrocompatibilité	La capacité pour une application de fonctionner sur des versions précédentes de la plateforme
Code-barres	Une représentation optique des données, lisible par une machine

Téléphone basique/de base	Un téléphone mobile avec une fonctionnalité minimale comme faire des appels, stocker des numéros de téléphone, envoyer des SMS, horloge et alarme.
Angle mort	Un emplacement sans réseau de télécommunications sans fil.
Mode de blocage/ne pas déranger [NDT : silencieux]	Un mode opérationnel d'appareils mobiles qui peut être activés par l'utilisateur pour supprimer certaines fonctionnalités - notifications courantes et appels vocaux.
Bluetooth	Une technologie de communication sans fil à courte portée.
Byte-code	Un ensemble d'instructions conçu pour une exécution efficace par un interprète logiciel. Aussi appelé code portable ou p-code.
Données cellulaires	Données transférées sur un réseau cellulaire.
Réseau cellulaire	Un réseau cellulaire est un réseau créé de plusieurs cellules indépendantes mais connectées.
Dispositif compagnon	Un appareil informatique conçu pour fonctionner en coopération avec un appareil intelligent dépendant.
Fréquence CPU	La fréquence d'horloge du processeur.
Framework de développement multiplateforme	un framework pour développer une application pour différentes plateformes en utilisant la même base de code.
CRUD	Mnémonique pour Create/Read/Update/Delete qui s'applique aux données.
Intégrité des données	L'exactitude et la cohérence des données sur l'ensemble de son cycle de vie, y inclus le stockage, le traitement, et la récupération.
Synchronisation des données	Le processus d'amener les données dans le même état à travers deux sources ou plus.
Validation des données	L'évaluation si les données sont correctes, précises, cohérentes et utiles.
Point mort	Voir « angle mort »
Fragmentation des appareils	La diversité des appareils disponibles, leur configuration matérielle unique et leur impact sur les applications et les utilisateurs
DPI/PPI	Acronyme pour points/pixels par pouce - un nombre exprimant la densité d'un affichage, que ce soit en points ou en pixels.
Émulateur	Une application logicielle qui imite le comportement du matériel.
Application d'entreprise	Une application créée pour être utilisée à l'interne au sein d'une organisation et non destinée à un usage public.
Mémoire externe	Une mémoire supplémentaire qui est ajoutée à l'appareil via une interface standard. Couramment les cartes SD sont les plus communes pour les téléphones mobiles.
Client lourd	Dans les applications client/serveur, un client qui a été conçu pour gérer une partie ou la plupart du traitement des données
Téléphone doté de fonctionnalités / feature phone	Une classe de téléphones mobiles qui fournit plus de fonctions qu'un téléphone de base, par exemple, un

	navigateur, mais ne fournit pas toutes les fonctionnalités d'un smartphone.
Fichier plat	Un fichier n'ayant pas de hiérarchie interne.
Mode avion	Un mode de fonctionnement spécial pour les appareils mobiles où les émetteurs radio sont désactivés afin d'éviter toute interférence avec les systèmes de fonctionnement et de communication de vol
Force touch	Une technologie développée par Apple Inc. qui permet aux trackpads et aux écrans tactiles de faire la distinction entre les différentes quantités de force appliquées sur leurs surfaces.
Application de premier plan	Une application qui est exécuté au premier plan de l'appareil pour l'interaction directe de l'utilisateur.
Application Freemium	Un modèle d'affaires dans lequel les utilisateurs ne paient rien pour télécharger l'application et se voient offrir des achats optionnels dans l'application.
Geste	Un certain modèle d'interaction, comme un pincement ou un balayage, pour activer les fonctions définies de l'appareil. Par exemple, un pincement est couramment utilisé pour zoomer sur l'écran de l'appareil intelligent.
Mondialisation	Voir internationalisation.
GPS	Acronyme pour Global Positioning System - dans le monde entier, un réseau de satellites envoie des signaux de temps. En incluant le signal d'au moins 3 satellites, un récepteur peut calculer sa position relative aux satellites par triangulation.
GSM	Acronyme pour GSM (Global System for Mobile Communications, à l'origine Groupe Spécial Mobile) est une norme élaborée par l'European Telecommunications Standards Institute (ETSI) pour décrire les protocoles des réseaux cellulaires numériques de deuxième génération utilisés par les appareils mobiles. Actuellement la norme la plus courante pour la communication mobile dans le monde.
Cellule GSM	Une partie d'un réseau GSM qui peut être identifiée par son identifiant cellulaire unique
Application hybride	Une application combinant les technologies natives et web. Habituellement, une application hybride utilise un cadre natif pour être installé sur l'appareil pour interagir avec les bibliothèques d'appareils ou autres. En outre, le contenu qui est affiché est reçu à partir d'un serveur Web.
I18N	Numéronyme (mot basé sur le nombre) pour l'internationalisation.
IDE	Environnement de développement intégré Une application logicielle qui fournit des installations complètes aux programmeurs informatiques pour le développement de logiciels.

Achats intégrés	Contenu et fonctionnalités supplémentaires disponibles directement à partir d'une application
Instruments	Un outil d'analyse et de test de performance inclus dans l'ensemble d'outils Xcode.
Mémoire interne	Une mémoire qui est incluse dans le matériel de l'appareil.
Internationalisation	Le processus de préparation d'une application pour supporter diverses versions localisées.
Interruption	Un événement qui se produit lors d'un autre événement.
Appareil IOT	Internet des objets. Un appareil ou, par exemple, un capteur d'intérêt connecté à Internet.
Jailbreaking	Une escalade des privilèges dans le but de supprimer les restrictions logicielles imposées par un système d'exploitation. Terme habituellement utilisé sur iOS. Semblable à rooting sur Android.
L10N	Numéronyme pour la localisation.
Mode paysage	L'orientation de l'appareil dans laquelle la largeur de l'écran est plus grande que la hauteur.
Bibliothèque	Une collection de ressources non volatiles utilisées par les programmes informatiques c'est-à-dire, les fonctions de l'application
Localisation	Le processus d'ajustement d'une application ou d'un produit à une certaine région par des actions telles que la traduction et l'ajustement de format.
Look and feel	Impression visuelle et émotionnelle de quelque chose.
Mnémonique	Une aide à la mémoire.
Mobile Application Mobile Testing	Tests des applications mobiles.
Type d'appareil mobile	Une classification des appareils mobiles par leurs caractéristiques de base. Les classes courantes incluent le téléphone de base, le téléphone doté de fonctionnalités, le smartphone, la phablette, la tablette, et le dispositif portable.
Émulateur mobile	Représentation virtuelle d'une plate-forme matérielle. Par exemple, l'émulateur Android est un matériel virtuel qui exécute une image Android OS réel. La même image OS pourrait être déployée sur le matériel et fonctionnera, car c'est le véritable OS.
OS mobile	Un système d'exploitation spécialement conçu pour les appareils mobiles.
Plateforme mobile	Un écosystème autour d'un système d'exploitation mobile, comprenant généralement des outils de développement, le système d'exploitation lui-même et un canal de distribution d'applications.

Espace mobile	Un terme encapsulant qui inclut tout ce qui concerne la technologie des appareils mobiles, du marché et de ses acteurs aux appareils et applications.
Simulateur mobile	Un environnement d'exécution virtuel. Par exemple, le simulateur iOS prétend être iOS, mais n'est en fait pas un véritable iOS.
Applications multiplateformes	Applications conçues et développées pour fonctionner sur plusieurs plateformes et utilisant la même base de code pour toutes les plateformes.
Multi-tier	Une approche de conception d'application backend où 2 serveurs ou plus fournissent des fonctionnalités spécialisées.
Multi-touch	Un type d'interaction avec un appareil utilisant divers événements tactiles en parallèle.
Application native	Une application spécialement développée pour une certaine plateforme, généralement en utilisant les APIs et les outils de développement spécifiques à la plateforme.
NFC	« Near Field Communication » - une technologie de communication radio à courte portée
Notification	Une annonce envoyée par l'appareil.
OCR	« Optical Character Recognition ». La reconnaissance des images de texte contenues dans une image électronique et sa conversion en texte codé à la machine.
Laboratoire sur site	Un laboratoire physiquement situé au même endroit que l'utilisateur du laboratoire.
Orientation	Le placement d'un objet dans le mobile couramment utilisé pour exprimer la façon dont l'appareil est utilisé. Il peut s'agir de paysage ou de portrait.
OTA	« Over the air ». Transmission de données par signaux radio, couramment utilisés pour désigner l'installation d'applications à un appareil directement à partir d'une source non connectée par câble.
Débordement [NDT : overflow]	Une situation où les données entrantes dépassent ce qui peut être géré.
Application payante	Une application monétisée en la vendant dans les app stores.
Persona	Un modèle/archétype pour un certain groupe d'utilisateurs.
Mode portrait	Une orientation de l'appareil dans laquelle la hauteur d'affichage est supérieure à la largeur.
Consommation d'énergie	La quantité d'énergie consommée.
Mode d'économie d'énergie	Un mode opérationnel d'appareils mobiles qui peut être activé par l'utilisateur ou l'appareil lui-même pour conserver l'énergie
État de charge	Un profil défini ou prédéfini par l'utilisateur en ce qui concerne la consommation d'énergie qui peut s'activer sur un appareil mobile.
Préférences	Les paramètres généraux de configuration de l'appareil ou de l'application qui peuvent être modifiés par l'utilisateur.

Application préinstallée	Une application mobile qui est installée par le fabricant de l'appareil. En général l'utilisateur n'est pas en mesure de désinstaller ces applications.
QR code	QR code (abrégé de « Quick Response » code) est la marque déposée d'un type de code à barres matricielle ou de code à barres bidimensionnelle.
Accès à distance de l'appareil	Interagir avec un appareil physiquement situé à un endroit différent de son utilisateur, habituellement sur Internet.
Conserver les données de l'application	Les données et le contenu générés par l'utilisateur de l'application sont conservés sur l'appareil lorsque l'application est désinstallée afin d'être accessible à d'autres applications ou lorsque la même application est réinstallée.
Rooting	Le processus d'accès root au système d'exploitation de l'appareil. Terme est généralement utilisé sur la plate-forme Android. Semblable à jailbreaking sur iOS.
Environnement d'exécution [NDT: run-time]	Mise en œuvre du modèle d'exécution. Aussi connu sous le nom d'exécution système.
Surface d'écran [NDT: screen real-estate]	La quantité d'espace fournie par l'écran.
Software development kit (SDK)	Un ensemble d'outils et de bibliothèques pour développer des logiciels pour une certaine plateforme.
Données sensibles	Données qui ont besoin d'une protection spéciale, telles que les mots de passe et les données personnelles.
Capteur	Un dispositif, module ou sous-système dont le but est de détecter des événements ou des changements dans son environnement et d'envoyer l'information à d'autres appareils électroniques, fréquemment un processeur informatique.
Session	Un événement défini dans le temps (boîte de temps).
Feuille de session	Un document pour évaluer et enregistrer une session de test.
side-loading	Le chargement/installation d'une application par un autre moyen qu'un magasin d'application.
Single-tier	Une approche de conception d'application backend dans laquelle un serveur unique fournit tous les services nécessaires pour une application.
Smartphone	Un ordinateur personnel portatif avec un système d'exploitation mobile et une connexion intégrée au réseau cellulaire mobile à large bande pour la voix, les SMS (service de messages courts ; souvent appelé textos) et la communication de données Internet.
clavier virtuel [NDT: soft keyboard]	Un clavier virtuel réalisé dans un logiciel, présenté à l'utilisateur sur un affichage
Store-and-forward	Une approche de synchronisation des données où les données sont stockées localement et transmises au serveur lorsqu'il existe une connexion réseau appropriée.

Communication synchrone	Méthode de transfert de données dans laquelle les flux de données sont envoyés (en amont) et reçus (en aval) à la même vitesse et espacés par des signaux de temps.
Tablette	Un type d'appareil mobile, couramment utilisé pour les appareils avec des écrans de 7"et plus.
Client léger	Dans les applications client/serveur, un client conçu pour être particulièrement petit de sorte que la majeure partie du traitement de données se produit sur le serveur.
Place de marché tierce	Une plateforme de distribution d'applications qui n'est pas gérée par un fournisseur de plateforme
Application basée sur la transaction	Une application où l'utilisateur paie à la transaction
Conflit de téléchargement [NDT : upload conflict]	Une erreur qui survient en essayant de télécharger un fichier qui est déjà présent à la destination.
Taille de fenêtre [NDT : viewport size]	Une taille d'écran virtuelle utilisée par le navigateur pour ajuster la mise en page à l'écran.
Virtual Private Network (VPN)	Une canal privé crypté via un réseau public.
Dispositif portable [NDT : wearable]	Un appareil informatisé porté sur soi comme une montre ou des lunettes.
Application Web	Une application hébergée sur le Web et utilisée via un navigateur
Xcode	Un environnement de développement intégré fourni par Apple pour développer des applications OSX et iOS.