Certified Tester Specialist

ISTQB® Mobile Application Testing Foundation Level

Syllabus

Versione 2019

Redatto da International Software Quality Institute (iSQI)

International Software Testing Qualifications Board

Gli autori di Certified Mobile Application Professional - Foundation level (CMAP-FL) syllabus - Jose Diaz, Rahul Verma, Tarun Banga, Vipul Kocher e Yaron Tsubery - hanno trasferito i diritti d'autore a ISTQB®. Questo syllabus è stato usato come base per creare il presente documento.

Questo documento è stato prodotto da un team di professionisti dell'International Software Testing Qualifications Board Mobile Application Testing Working Group.

Vipul Kocher (presidente), Piotr Wicherski (vicepresidente), José Díaz, Matthias Hamburg, Eran Kinsbruner, Björn Lemke, Samuel Ouko, Tal Pe'er, Ralf Pichler, Lloyd Roden, Nils Röttger, Angelina Samaroo, Yaron Tsubery

Cronologia revisioni

Versione	Data	Indicazioni
Alpha	11 maggio 2018	Versione alpha
Beta	27 gennaio 2019	Versione beta
GA	28 marzo 2019	Versione GA
V2019	3 maggio 2019	Versione ISTQB®

Sommario

Riconoscimenti

Questo documento è stato prodotto da un team di professionisti dell'International Software Testing Qualifications Board Mobile Application Testing Working Group.

Vipul Kocher (chair), Piotr Wicherski (vice-chair), José Díaz, Matthias Hamburg, Eran Kinsbruner, Björn Lemke, Samuel Ouko, Tal Pe'er, Ralf Pichler, Lloyd Roden, Nils Röttger, Angelina Samaroo, Yaron Tsubery

La suddetta squadra ringrazia il team di revisione per i suoi suggerimenti e contributi.

Le seguenti persone hanno partecipato alla revisione, al commento o al voto di questo 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 Sótér, Marco Sogliani, Michael Stahl, Chris Van Bael, Paul Weymouth, Salinda Wickramasinghe, Minghui Xu

Il presente documento è stato ufficialmente pubblicato da ISTQB® il 3 maggio 2019.

0. Premessa

0.1 Scopo del documento

Il presente syllabus costituisce la base per la certificazione di livello Foundation in Mobile Application Testing. ISTQB® fornisce il syllabus come segue:

1. Ai National Boards, per consentire la traduzione nella loro lingua locale e accreditare gli enti di formazione. I National Boards possono adattare il syllabus alle loro esigenze linguistiche particolari e modificare i riferimenti per adattarli alle pubblicazioni locali.
2. Agli Exam Boards, per consentire la formulazione delle domande d'esame nella loro lingua locale adattate agli obiettivi di apprendimento per ciascun syllabus.
3. Agli enti di formazione, per consentire la produzione del materiale didattico e determinare metodi di insegnamento appropriati.
4. Ai candidati alla certificazione per la preparazione all'esame (sia se prendono parte a un corso di formazione sia se studiano individualmente).
5. Alla comunità internazionale di ingegneria del software e dei sistemi, per progredire nella professione di testing di software e sistemi e come base per libri e articoli.

ISTQB® può consentire ad altri organismi di utilizzare il presente syllabus per altri scopi, previa autorizzazione scritta.

0.2 La Certified Foundation Level Mobile Application Testing

Questa certificazione di livello Foundation è rivolta a chiunque sia coinvolto nei test di software che desideri ampliare la propria conoscenza in mobile application testing o a chiunque desideri iniziare una carriera specialistica in questo ambito.

Le informazioni sul mobile application testing descritte nel syllabus ISTQB® Certified Tester Foundation Level [ISTQB_CTFL_2018] sono state prese in considerazione nella creazione del presente syllabus.

0.3 Business outcomes

Questa sezione elenca i Business outcomes attesi da un candidato che ha conseguito la certificazione Foundation Level Mobile Application Testing.

MAT-01 Comprendere e rivedere gli elementi chiave aziendali e tecnologici per le app mobili al fine di creare una strategia di test.

MAT-02 Identificare e comprendere le principali sfide, i rischi e le aspettative associati al test di un'applicazione mobile.

MAT-03 Applicare tipi di test e livelli specifici per le applicazioni mobili.

MAT-04 Applicare tipi di test comuni, come quelli menzionati in [ISTQB_CTFL_2018], nel contesto specifico dei dispositivi mobili.

MAT-05 Eseguire le attività richieste specificatamente per i test delle applicazioni mobili come parte delle attività principali descritte nel processo di test ISTQB® .

MAT-06 Identificare e utilizzare ambienti e strumenti adeguati per i test delle applicazioni mobili.

MAT-07 Comprendere i metodi e gli strumenti utilizzati specificamente per supportare l'automazione dei test delle applicazioni mobili.

0.4 Obiettivi di apprendimento oggetto di esame

Questi obiettivi di apprendimento supportano i risultati aziendali e sono la base per l'esame da superare per ottenere la certificazione Foundation Level Mobile Application Testing. Gli obiettivi di apprendimento sono assegnati a un livello cognitivo di conoscenza (K-Level).

Un K-Level, o livello cognitivo, viene utilizzato per classificare gli obiettivi di apprendimento in base alla tassonomia rivista da Bloom [Anderson 2001]. ISTQB® utilizza questa tassonomia per progettare gli esami rispetto ai suoi syllabi.

Il presente syllabus considera tre diversi K-Level (da K1 a K3). Per ulteriori informazioni, consultare il capitolo 7.

0.5 Livelli di competenza pratica

Mobile Application Testing Foundation Level introduce il concetto di obiettivi pratici incentrati su abilità e competenze pratiche.

Le competenze possono essere raggiunte eseguendo esercizi pratici, come quelli indicati nel seguente elenco non esaustivo:

- Esercizi per obiettivi di apprendimento di livello K3 eseguiti utilizzando carta e penna o software di elaborazione testi, come avviene per vari syllabi ISTQB® esistenti.
- Impostazione e utilizzo di ambienti di test.
- Test delle applicazioni su dispositivi virtuali e fisici.
- Utilizzo di strumenti su desktop e/o dispositivi mobili per testare o assistere nel test di attività correlate come installazione, esecuzione di query, registrazione, monitoraggio, cattura di schermate, ecc.

I seguenti livelli si applicano agli obiettivi pratici:

- H0: può includere una demo live di un esercizio o un video registrato. Dal momento che non viene eseguito dal corsista, non è strettamente un esercizio.
- H1: esercizio guidato. I corsisti seguono una sequenza di passaggi eseguiti dall'insegnante.
- H2: esercizio con suggerimenti. Al corsista viene dato un esercizio con suggerimenti pertinenti che gli consentano di risolverlo entro un determinato periodo di tempo.
- H3: esercizi non guidati senza suggerimenti.

Raccomandazioni:

- Gli obiettivi di apprendimento K1 in genere usano il livello H0 e H1 o H2 quando la situazione lo richiede.
- Gli obiettivi di apprendimento K2 in genere utilizzano i livelli H1 o H2 e H0 o H3 quando la situazione lo richiede.

Gli obiettivi di apprendimento K3 utilizzano generalmente i livelli di H2 o H3, sebbene non sia sempre necessario esercitarsi per un obiettivo formativo K3. Se l'installazione è complessa o richiede troppo tempo, utilizzare il livello H0.

0.6 L'esame

L'esame per la certificazione Mobile Application Testing Foundation Level si baserà su questo syllabus. Le risposte alle domande d'esame possono richiedere conoscenze basate su più di una sezione di questo syllabus. Tutte le sezioni del syllabus sono tema di esame, a eccezione dell'introduzione e delle appendici. Standard, libri e altri syllabi ISTQB® sono inclusi come riferimenti, ma il loro contenuto non è tema di esame, al di là di quanto riassunto nel presente syllabus.
L'esame per la certificazione Mobile Application Testing Foundation Level si basa su domande a risposta multipla. È costituito da 40 domande. Per superare l'esame, è necessario rispondere correttamente ad almeno il 65% delle domande (ovvero 26 domande). Gli obiettivi pratici e gli esercizi non saranno tema d'esame.
Gli esami possono essere sostenuti nell'ambito di un corso di formazione accreditato o in modo indipendente (ad es. presso un centro d'esame o in un esame pubblico). Il completamento di un corso di formazione accreditato non rappresenta un prerequisito per l'esame.
Chiunque desideri presentarsi all'esame senza aver frequentato un corso seguito presso un centro di formazione accreditato dovrebbe leggere le linee guida sulle competenze come indicato nel documento Accreditation and Competence Guidelines (Linee guida per l'accreditamento e le competenze) [CTFL-MAT-2019-Accreditation-and-Competence-Guidelines.pdf] e provare a eseguire autonomamente questi esercizi pratici. Ciò li aiuterà ad acquisire le competenze che un centro di formazione accreditato dovrebbe fornire. Si noti che ciò non incide sull'esame per la certificazione Mobile Application Testing Foundation Level poiché esso si basa solo su questo syllabus e sui suoi obiettivi di apprendimento.

0.7 Tempi di preparazione consigliati

In questo syllabus è stato definito un tempo minimo di preparazione per ciascun obiettivo formativo. Il tempo totale per ciascun capitolo è indicato nell'intestazione del capitolo stesso.

Gli enti di formazione dovrebbero notare che altri syllabi ISTQB® applicano un approccio a "tempo standard" che assegna tempi fissi in base al K-Level. Il syllabus di Mobile Application Testing non applica questo schema rigorosamente. Di conseguenza, agli enti di formazione viene data un'indicazione più flessibile e realistica dei tempi minimi di preparazione per ciascun obiettivo formativo.

0.8 Condizioni di ammissione

La certificazione ISTQB® Foundation Level deve essere conseguita prima di sostenere questo esame.

0.9 Fonti di informazione

I termini utilizzati nel syllabus sono definiti nel Glossario dei termini ISTQB® utilizzati in Software Testing [ISTQB_GLOSSARY].

Il capitolo 6 contiene un elenco di libri e articoli consigliati sul Mobile Application Testing.

1. Mobile World - Fattori Aziendali e Tecnologici - 175 minuti

Parole chiave

analisi del rischio, mitigazione del rischio, testing basato sul rischio, strategia di test

Obiettivi di Apprendimento per i Fattori Aziendali e Tecnologici

1.1 Dati di Mobile Analytics

MAT-1.1.1 (K2) Descrivere come utilizzare dati di mobile analytics come input per la strategia di test e il piano di test.

HO-1.1.1 (H3) In base ai dati raccolti da una o più fonti di dati di analytics (posizione geografica, piattaforma, versione del sistema operativo e distribuzione del tipo di dispositivo), selezionare i tipi di dispositivo da testare e la corrispondente priorità.

Nota: HO-1.1.1 e HO-1.7.1 (sotto) possono essere combinati.

1.2 Modelli di Business per App Mobili

MAT-1.2.1 (K2) Distinguere tra vari modelli di business per applicazioni mobili.

1.3 Tipi di Dispositivi Mobili

MAT-1.3.1 (K1) Ricordare i diversi tipi di dispositivi mobili.

1.4 Tipi di Applicazioni Mobili

MAT-1.4.1 (K2) Distinguere tra diversi tipi di applicazioni mobili.

1.5 Architettura delle Applicazioni Mobili

MAT-1.5.1 (K2) Distinguere tra tipi di architettura generali di applicazioni mobili.

1.6 Strategia di Test per App Mobili

MAT-1.6.1 (K3) Applicare caratteristiche e specifiche del mercato mobile nella preparazione di una strategia di test.

1.7 Sfide del Mobile Application Testing

MAT-1.7.1 (K2) Fornire esempi delle sfide associate ai test di applicazioni mobili.

HO-1.7.1 (H1) Raccogliere dati di mercato come quote di mercato dei dispositivi o dei sistemi operativi di una data regione. Raccogliere dati per dimensioni e densità dello schermo. Creare un elenco di cinque dispositivi e calcolare la copertura di mercato prevista per questo elenco.

Nota: HO-1.1.1 (vedi sopra) e HO-1.7.1 possono essere combinati.

1.8 Rischi nel Mobile Application Testing

MAT-1.8.1 (K2) Descrivere come mitigare i rischi specifici delle applicazioni mobili.

1.1 Dati di Mobile Analytics

Esistono molti stakeholder nel mondo mobile tra cui produttori, fornitori di piattaforme, di sistemi operativi, di dati di mercato, di strumenti e, naturalmente, sviluppatori e tester di applicazioni.

Al fine di contribuire efficacemente alle discussioni sulla pianificazione e all'analisi dei test, un tester di applicazioni mobili deve conoscere i seguenti fattori:

- Le implicazioni commerciali della distribuzione di piattaforme
- Download di applicazioni per piattaforma
- La quantità e la distribuzione delle versioni del sistema operativo
- La distribuzione sul mercato di vari tipi di dispositivi, comprese le variazioni in base alla posizione geografica
- Dimensioni e risoluzioni dello schermo differenti
- I vari metodi di input
- Tipi di fotocamera

Esistono diverse fonti di informazioni per i fattori di cui sopra, sia gratuite che commerciali. Queste includono StatCounter GlobalStats [URL1], i fornitori stessi del sistema operativo e altre fonti di terze parti.

I dati di mobile analytics vengono utilizzati per selezionare un portfolio di dispositivi per l'esecuzione dei test che sia appropriato per il mercato di destinazione. I test vengono eseguiti su questo portfolio per testare l'app su un dispositivo in base all'importanza del dispositivo. I dati relativi ai dispositivi e alle loro eventuali funzioni speciali possono anche essere utilizzati per progettare test specifici per un tipo di dispositivo. Ad esempio, un dispositivo con sensore per il battito cardiaco potrebbe richiedere test speciali.

1.2 Modelli di Business per App Mobili

Esistono diversi modelli che possono essere utilizzati per monetizzare il lavoro svolto nella creazione di applicazioni mobili. Questi includono, ma non sono limitati a: applicazioni Freemium, advertisement-based, transaction-based, fee-based ed enterprise. Inoltre, alcuni acquisti in-app possono essere applicati ad alcuni di questi modelli.

Vi sono alcuni vantaggi e svantaggi per ciascuno di questi approcci e il tester deve tenere presente il modello di business durante il testing dell'applicazione mobile.

In un modello Freemium, le applicazioni sono generalmente gratuite ma gli utenti devono pagare se necessitano di funzionalità aggiuntive. Tali applicazioni devono fornire funzionalità sufficienti per essere attraenti per gli utenti, fornendo al contempo funzioni avanzate per le quali vi sia un gran numero di utenti disposto a pagare.

Le applicazioni advertisement-based visualizzano annunci pubblicitari sullo schermo mentre gli utenti interagiscono con le applicazioni stesse. Questa strategia per la generazione di entrate è più efficace se le applicazioni vengono utilizzate per periodi di tempo relativamente lunghi. I designer di interfacce utente devono fare attenzione quando visualizzano gli annunci. Tali annunci devono essere ben in vista senza nascondere parti essenziali dell'applicazione e bisogna garantire che gli utenti non siano eccessivamente distratti da non apprezzare l'utilizzo dell'applicazione.

Le applicazioni transaction-based addebitano agli utenti una transazione, una commissione fissa, una percentuale del valore della transazione o simili. Questo tipo di modello di business è adatto solo per un numero limitato di applicazioni ed è generalmente applicato ad app aziendali e finanziarie come i portafogli mobili.

Le applicazioni fee-based richiedono agli utenti di pagare per il download e l'installazione dell'applicazione stessa. È necessario prendere in considerazione la scelta di un modello commerciale fee-based poiché esiste un gran numero di opzioni free o freemium per la maggior parte dei tipi di applicazione. La probabilità che gli utenti acquistino tale app aumenta se essa fornisce funzionalità o usabilità eccellenti o quando non sono disponibili applicazioni concorrenti.

Le applicazioni free ed enterprise non richiedono alcun pagamento da parte dei rispettivi utenti. Le applicazioni enterprise sono sviluppate per essere usate all'interno dell'organizzazione e forniscono un'interfaccia per accedere ai servizi forniti. Esistono molte di queste app rese disponibili da organizzazioni come banche o società di e-commerce. Non si concentrano generalmente sulla monetizzazione dell'app stessa, ma consentono di generare entrate indirizzando gli utenti ai servizi forniti dalle organizzazioni.

1.3 Tipi di Dispositivi Mobili

Sono disponibili numerosi dispositivi mobili che supportano diversi tipi di applicazioni.

I dispositivi comuni includono:

- Telefoni base (basic phone)
- Telefoni con funzionalità aggiuntive (feature phone)
- Smartphone
- Tablet
- Dispositivi associati - inclusi dispositivi indossabili e alcuni dispositivi IoT (Internet of Things).

Durante il testing, è necessario tenere presente che ogni tipo di dispositivo ha caratteristiche specifiche per esigenze particolari.

I telefoni base vengono utilizzati solo per chiamate e SMS e hanno pochissime app e giochi integrati. Non è possibile installare app o navigare in Internet.

I telefoni con funzionalità aggiuntive offrono un supporto limitato per le app e la loro installazione. Forniscono l'accesso a Internet tramite un browser integrato e possono disporre di hardware aggiuntivi come le fotocamere.

Gli smartphone forniscono ai telefoni diversi sensori. Il sistema operativo supporta funzionalità quali l'installazione delle applicazioni, il supporto multimediale e la navigazione.

I tablet sono simili agli smartphone ma sono fisicamente più grandi. In genere vengono utilizzati quando serve o si desidera uno schermo più grande e possono anche supportare una maggiore durata della batteria.

I dispositivi associati e alcuni dispositivi IoT sono dispositivi forniti di computer, comunemente utilizzati insieme a uno smartphone o un tablet per estendere le funzionalità disponibili o consentire l'accesso ai dati sul telefono o sul tablet in modo più conveniente.

I dispositivi indossabili sono dispositivi che possono essere indossati dai consumatori. Questi possono fungere da accompagnamento ai dispositivi esistenti o funzionare in modo indipendente. Orologi e cinturini per il fitness sono esempi comuni di dispositivi indossabili.

1.4 Tipi di Applicazioni Mobili

Esistono tre tipi principali di applicazioni mobili:

- Nativa
- Browser-based (app web)
- Ibrida

Ognuna di esse presenta certi vantaggi e svantaggi, per cui è necessario prendere una decisione a livello aziendale prima di avviare lo sviluppo.

Le applicazioni native sono sviluppate utilizzando kit di sviluppo software (SDK), strumenti di sviluppo, sensori e funzionalità specifici per la piattaforma. Vengono scaricate, installate e aggiornate dagli store dei fornitori. Potrebbe essere necessario testare queste app su tutti i dispositivi supportati.

Le applicazioni native in genere forniscono prestazioni migliori, possono sfruttare appieno le funzionalità della piattaforma e soddisfare le aspettative per la piattaforma per cui sono state sviluppate. Il loro costo di sviluppo è generalmente più elevato e possono richiedere ulteriori sfide, come l'uso di più piattaforme e l'installazione e il testing su un gran numero di dispositivi.

Le applicazioni browser-based sono accessibili tramite un browser mobile. Dal momento che utilizzano le tecnologie e i browser di sviluppo tipici del web, il supporto su piattaforme multiple è semplice e il costo di sviluppo è generalmente inferiore.

Esistono quattro modi principali con cui vengono create le applicazioni web mobili:

- Versioni specifiche per dispositivi mobili di siti web e applicazioni (detti anche m(dot) site). Di solito ciò significa che quando un browser mobile si rivolge all'applicazione, viene fornita una versione mobile dell'applicazione stessa. Ad esempio, facebook.com reindirizza a m.facebook.com quando vi si accede da un dispositivo mobile.
- Le app web ‘responsive’ assicurano che il design si adatti al fattore di forma e alle dimensioni dello schermo, di solito espressi come view port.
- Le app web ‘adaptive’ regolano il design in base ad alcune dimensioni predefinite. Esistono diversi design per queste dimensioni e le funzionalità disponibili per l'utente sono spesso regolabili.
- Le app web ‘progressive’ consentono di creare collegamenti a pagine web specifiche nella schermata principale del dispositivo mobile. Sembrano app native e talvolta possono persino funzionare offline.

Le app web per dispositivi mobili vengono create utilizzando tecnologie web comuni, il che generalmente le rende più facili da sviluppare e gestire rispetto alle app native e ibride. Tuttavia, potrebbero non essere così ricche di funzionalità come le app native o ibride e avere un accesso limitato alle API (Application Programming Interface) native della piattaforma. Anche l'accesso ai sensori mobili è limitato. Non è necessario il test di installabilità sui dispositivi, ma è richiesto quello di compatibilità del browser.

Le applicazioni ibride sono una combinazione di app native e app web. Usano un wrapper di app nativa che contiene una vista web per eseguire un'applicazione web all'interno di un'app

nativa. Queste app vengono scaricate dagli store dei fornitori e possono accedere a tutte le funzionalità del dispositivo. Sono relativamente facili da sviluppare, aggiornare e gestire senza aggiornare l'app installata sul dispositivo. Le competenze richieste per lo sviluppo di queste app sono quasi le stesse dello sviluppo web. I possibili punti deboli di queste app includono problemi di prestazioni dovuti all'uso di un wrapper e possibili scostamenti rispetto al “look and feel” previsto, a causa di caratteristiche specifiche della piattaforma.

Le app native e ibride sono installate fisicamente su un dispositivo e sono quindi sempre disponibili per l'utente, anche quando il dispositivo non ha una connessione Internet. Al contrario, le applicazioni browser-based richiedono l'accesso a Internet.

Alcune applicazioni sono preinstallate sul dispositivo mobile e altre possono essere installate tramite vari canali di distribuzione, come Apple App Store, Google Play Store, app store aziendali (disponibili solo all'interno della rete aziendale) e mercati di app di terze parti.

Il testing di ciascuno di questi tipi di applicazione può richiedere un approccio diverso. I parametri da considerare includono:

- Diversi tipi di dispositivi da supportare
- Caratteristiche del sensore e del dispositivo da utilizzare
- Disponibilità in varie condizioni di rete
- Installabilità, compatibilità, efficienza delle prestazioni e usabilità

1.5 Architettura delle Applicazioni Mobili

Esistono diverse soluzioni per la progettazione di un'applicazione mobile.

Alcune delle considerazioni nella scelta di una particolare architettura o decisione progettuale includono:

- Pubblico destinatario
- Tipo di applicazione
- Supporto di varie piattaforme mobili e non mobili
- Esigenze di connettività
- Esigenze di archiviazione dei dati
- Connessioni ad altri dispositivi, compresi gli apparecchi IoT

Le decisioni sull'architettura includono:

- Architettura lato client come thin o fat client
- Architettura lato server come single o multi-tier
- Tipo di connessione come Wi-Fi, rete dati cellulare, Near Field Communication (NFC), Bluetooth
- Metodi di sincronizzazione dei dati come comunicazioni store-and-forward, push and pull, sincrone e asincrone

Le app thin client non contengono codice dell'applicazione personalizzato per il dispositivo e fanno un uso minimo delle funzionalità del sistema operativo mobile. Queste app in genere

utilizzano il browser web come front-end e JavaScript come linguaggio per l'implementazione della logica lato client.

Le applicazioni client thick/fat possono avere più livelli di codice e utilizzare le funzionalità del sistema operativo mobile. Si tratta in genere di applicazioni native o ibride.

Le architetture lato server includono le seguenti possibilità:

- Le architetture single-tier sono monolitiche e hanno tutti i server sulla stessa macchina. Sono meno scalabili e più difficili da proteggere.
- Le architetture multi-tier distribuiscono i componenti lato server su varie unità. Le architetture two-tier coinvolgono web server e database server separati, mentre le architetture a three-tier includono anche un application server. Le architetture multi-tier consentono la separazione delle responsabilità, forniscono la specializzazione del database e offrono una migliore flessibilità, scalabilità e sicurezza. Tuttavia, possono essere significativamente più costose da sviluppare, gestire e ospitare rispetto alle architetture single-tier.

Esistono vari metodi di connessione. Un dispositivo mobile potrebbe essere collegato al server tramite tipi di connessione come il Wi-Fi o connessioni dati cellulari come il 2G, 3G, 4G e 5G. Le applicazioni mobili in genere funzionano in una delle tre modalità seguenti:

- Le app mai connesse (never-connected app) funzionano offline e non devono essere connesse. Una semplice calcolatrice è un esempio di tali app.
- Le app sempre connesse (always-connected app) richiedono una connessione di rete permanente durante il funzionamento. Tutte le applicazioni web mobili rientrano in questa categoria, sebbene alcune possano funzionare in modo limitato quando sono parzialmente connesse.
- Le app parzialmente connesse (partially-connected app) richiedono una connessione per attività come il trasferimento di dati, ma possono funzionare per lunghi periodi di tempo senza connessione.

La sincronizzazione dei dati tra il client e il server può essere condotta nelle seguenti modalità:

- La modalità continua è quella in cui i dati vengono trasferiti non appena vengono inviati.
- La modalità store-and-forward è quella in cui i dati possono essere archiviati localmente prima di essere trasferiti, specialmente quando non è disponibile alcuna connettività.

Il trasferimento dei dati può essere condotto con i seguenti due approcci:

- Il trasferimento di dati sincrono viene eseguito quando la funzione chiamante attende il completamento della funzione chiamata prima di tornare.
- Il trasferimento di dati asincrono viene eseguito quando la funzione del server chiamato termina immediatamente, elabora i dati in background e richiama la funzione del client chiamante una volta completata l'attività. Ciò offre agli utenti un maggiore controllo. Tuttavia, l'implementazione del meccanismo di handshake

aumenta la complessità relativa alla disponibilità del client o della rete quando il server avvia la callback.

1.6 Strategia di Test per App Mobili

La creazione di una strategia di test per dispositivi mobili richiede che il tester tenga conto di tutti i parametri elencati finora in questo capitolo. Inoltre, devono essere considerati anche i rischi discussi in questa sezione e le sfide descritte nel paragrafo 1.7.

I rischi comuni sono, ad esempio:

- Senza conoscere i dati di proliferazione dei dispositivi in una particolare posizione geografica, non è possibile scegliere i dispositivi su cui l'app deve essere testata in modo sostenibile.
- Senza conoscere il tipo di modello di business, non è possibile verificare se il comportamento dell'applicazione sia adatto per quel modello di business.

La creazione di una strategia di test per le applicazioni mobili deve inoltre considerare i seguenti rischi e sfide specifici:

- La varietà di dispositivi mobili con difetti specifici dei dispositivi stessi.
- La disponibilità di dispositivi in-house o tramite l'uso di laboratori di test esterni.
- L'introduzione di nuove tecnologie, dispositivi e/o piattaforme durante il ciclo di vita dell'applicazione.
- L'installazione e l'aggiornamento dell'app stessa tramite vari canali, che permettano il mantenimento dei dati e delle preferenze dell'app.
- I problemi della piattaforma che potrebbero influire sull'applicazione.
- La copertura della rete e il relativo impatto sull'app in un contesto globale.
- La capacità di eseguire test utilizzando le reti di vari fornitori di servizi.
- L'uso di emulatori/simulatori mobili e/o dispositivi reali per livelli e tipi di test specifici.

Queste sfide sono descritte in maggior dettaglio nella sezione 1.7.

La strategia di test tiene conto dei rischi e delle sfide. Per esempio:

- La strategia di test può specificare l'uso di emulatori/simulatori mobili nelle prime fasi di sviluppo e di dispositivi reali nelle fasi successive. Esistono solo alcuni tipi di test che possono essere eseguiti sugli emulatori/simulatori mobili. Ulteriori informazioni sono descritte nella sezione 4.3.
- La strategia di test può considerare la sfida rappresentata da un gran numero di dispositivi diversi adottando uno dei seguenti approcci:
    - Approccio a piattaforma singola: ridurre l’ambito a un solo tipo di dispositivo, una versione del sistema operativo, un operatore telefonico e un tipo di rete.

- Approccio a piattaforme multiple: ridurre l'ambito a una selezione rappresentativa di dispositivi e sistemi operativi utilizzati dalla maggior parte dei clienti nel mercato di destinazione, sulla base del traffico mobile o di altri dati di analytics.
- Approccio di massima copertura: coprire tutte le versioni del sistema operativo, i dispositivi, i produttori, gli operatori e i tipi di rete. Si tratta sostanzialmente di test esaustivi, che di solito non sono economicamente sostenibili, soprattutto se si considera la moltitudine di dispositivi e versioni del sistema operativo sul mercato.

- La strategia di test può considerare la sfida posta dalla non disponibilità di dispositivi, reti o condizioni reali utilizzando risorse esterne, come:
  - Servizi di accesso ai dispositivi remoti. Questo è un modo per accedere a dispositivi sul web che altrimenti non sono di proprietà.
  - Servizi di crowd testing. Questo è un modo per accedere a un enorme gruppo di volontari e ai loro dispositivi.
  - Reti personali come amici e colleghi, utilizzando la propria rete sociale privata.
  - Ricerca di bug. Consiste in un evento di test ludicizzato che utilizza volontari dell'azienda o del pubblico in generale.

Oltre ai livelli di test descritti in [ISTQB_CTFL_2018], la strategia di test considera anche i tipi comuni di test utilizzati per le applicazioni mobili (vedere la sezione 3.1) e tutti i livelli aggiuntivi di test richiesti (vedere la sezione 3.2).

1.7 Sfide del Mobile Application Testing

Nel mondo delle applicazioni mobili esistono molte altre sfide non comuni o non critiche nel software desktop o server. I tester devono essere consapevoli di queste sfide e di come potrebbero avere un impatto sul successo dell'applicazione.

Le sfide comuni nel mondo delle applicazioni mobili includono:

- Piattaforme multiple e frammentazione del dispositivo: più tipi e versioni di SO, dimensioni dello schermo e qualità del display.
- Differenze di hardware nei vari dispositivi: vari tipi di sensori e difficoltà nella simulazione delle condizioni di test per risorse CPU e RAM limitate.
- Varietà di strumenti di sviluppo software richiesti dalle piattaforme.
- Differenza di design dell'interfaccia utente e aspettative di esperienza utente (UX) sulle piattaforme.
- Più tipi di rete e provider.
- Dispositivi che richiedono molte risorse.
- Vari canali di distribuzione per le app.

- Diversi utenti e gruppi di utenti.
- Vari tipi di app con diversi metodi di connessione.
- Elevata visibilità del feedback derivante da bug che hanno un forte impatto sugli utenti e che possono facilmente comportare la pubblicazione di feedback sui marketplace online.
- Pubblicazioni sui marketplace che richiedono cicli di approvazione aggiuntivi per la pubblicazione da parte dei proprietari di tali marketplace, come Google Play o Apple App Store.
- Indisponibilità di dispositivi appena lanciati che richiedono l'uso di emulatori/simulatori mobili

Queste sfide implicano, ad esempio:

- Numerose combinazioni da testare.
- Numerosi dispositivi necessari per i test, con conseguente aumento dei costi.
- La necessità di garantire compatibilità all'indietro (backward compatibility) per eseguire l'applicazione su versioni precedenti della piattaforma.
- Nuove funzionalità rilasciate in ogni versione del sistema operativo.
- Linee guida da prendere in considerazione per le varie piattaforme.
- CPU che richiedono molte risorse e quantità limitata di memoria e spazio di archiviazione.
- Variabilità della larghezza di banda e del jitter con le varie reti.
- Modifiche delle velocità di upload e download disponibili in base ai piani dati.

I seguenti due esempi illustrano le sfide comuni e il loro potenziale impatto:

- Dispositivi diversi hanno differenti tipi di sensori e i test devono tenerne conto. Ogni nuovo sensore aggiunto all'hardware potrebbe richiedere ulteriori test di compatibilità all'indietro.
- Alcune delle sfide della rete possono essere affrontate in modo appropriato, anche in condizioni di rete variabili, utilizzando adeguate strategie di memorizzazione nella cache e di prefetching. Tuttavia, ciò ha un costo; un gran numero di connessioni aperte può influire sulle prestazioni del lato server poiché la maggior parte delle app mantiene l'utente connesso sul server.

1.8 Rischi nel Mobile Application Testing

Le sfide menzionate nella sezione 1.7 possono verificarsi singolarmente o in combinazione con altre. Ciò può comportare ulteriori rischi per un'applicazione mobile.

Un tester deve essere in grado di contribuire all'analisi del rischio del prodotto. I metodi comuni di analisi e mitigazione del rischio, come discusso in [ISTQB_CTFL_2018], capitolo 5.5, possono anche essere applicati nel contesto mobile. Inoltre, esistono i seguenti rischi specifici per dispositivi mobili e le corrispondenti strategie di mitigazione:

Rischio	Possibile mitigazione
Frammentazione del mercato	Effettuare una selezione appropriata di dispositivi per l'esecuzione dei test, ad esempio scegliere di svolgere i test sui dispositivi più comunemente utilizzati.
Costo del supporto di più piattaforme	Eseguire delle analisi per comprendere le piattaforme più utilizzate, evitando così il testing di quelle non più rilevanti.
Introduzione di nuove tecnologie, piattaforme e dispositivi	Utilizzare le versioni di pre-produzione di tali tecnologie.
Mancanza di disponibilità dei dispositivi per l'esecuzione dei test	Utilizzare servizi di accesso ai dispositivi remoti o servizi di crowd testing.
Rischi derivanti dai modelli di utilizzo previsti delle applicazioni mobili usate in prossimità del rilascio	Applicare approcci di test appropriati, come ad esempio il field testing.

2. Tipi di Test delle Applicazioni Mobili - 265 minuti

Parole chiave

coesistenza, compatibilità, connettività, compatibilità cross-browser, interoperabilità, sistema sotto test (SUT), tipo di test, usabilità

Obiettivi di Apprendimento per i Tipi di Test delle Applicazioni Mobili

2.1 Testing di Compatibilità con l'Hardware del Dispositivo

MAT-2.1.1 (K2) Descrivere le caratteristiche e l'hardware specifici del dispositivo che devono essere considerati per i test.

HO-2.1.1 (H1) Testare un'app con diverse funzionalità del dispositivo mobile mentre il sistema sotto test (SUT) è in uso per verificare il corretto funzionamento del SUT.

MAT-2.1.2 (K3) Preparare i test per la compatibilità dell'app con dimensioni, proporzioni e densità dello schermo.

HO-2.1.2 (H3) Testare un'app su diversi dispositivi mobili (virtuali o fisici) per mostrare l'impatto della risoluzione e delle dimensioni dello schermo sull'interfaccia utente dell'app.

MAT-2.1.3 (K2) Descrivere come i test possono mostrare i potenziali effetti del surriscaldamento del dispositivo sul sistema sotto test.

MAT-2.1.4 (K1) Ricordare i diversi tipi di test mirati a eseguire verifiche sui vari sensori di input utilizzati nei dispositivi mobili.

MAT-2.1.5 (K1) Ricordare i test da eseguire su vari metodi di input.

HO-2.1.5 (H0) Testare un'app per vari tipi di input, inclusi test relativi alla tastiera con più tastiere installate, test relativi a gesture e (se del caso) test relativi alla fotocamera.

MAT-2.1.6 (K2) Descrivere in che modo i test possono rivelare problemi all'interfaccia utente quando si cambia l'orientamento dello schermo.

HO-2.1.6 (H3) Testare un'applicazione per verificare l'effetto della modifica di orientamento dello schermo sulla funzionalità dell'app, inclusa la conservazione dei dati e la correttezza dell'interfaccia utente.

MAT-2.1.7 (K3) Preparare i test per un'app usando gli interrupt comuni del dispositivo mobile.

HO-2.1.7 (H3) Testare un'app con diversi interrupt del dispositivo mobile mentre l'applicazione è in uso.

MAT-2.1.8 (K3) Preparare i test per modificare le autorizzazioni di accesso alle funzionalità del dispositivo richieste dall'app.

HO-2.1.8 (H3) Testare la gestione delle autorizzazioni di un'app, autorizzando e negando le autorizzazioni richieste e osservando il comportamento quando le impostazioni di cartelle e sensori vengono negate durante l'installazione o modificate dopo l'installazione stessa.

MAT-2.1.9 (K3) Preparare i test per verificare l'impatto di un'app sul consumo energetico di un dispositivo e l'impatto del suo stato di alimentazione sull'app.

HO-2.1.9 (H3) Testare un'app con diversi livelli di carica della batteria per rilevare i dati di consumo e stabilire le prestazioni in condizioni di batteria quasi scarica e scarica completamente.

2.2 Testing delle Interazioni tra App e Software del Dispositivo

MAT-2.2.1 (K3) Preparare i test per la gestione delle notifiche da parte del sistema sotto test.

HO-2.2.1 (H2) Verificare l'effetto della ricezione di notifiche quando un'app è in foreground e in background. Testare l'effetto della modifica delle impostazioni di notifica sulla funzionalità dell'app.

MAT-2.2.2 (K2) Descrivere come i test possono verificare la corretta funzionalità dei collegamenti ad accesso rapido.

HO-2.2.2 (H3) Testare un'app per la funzionalità di scelta rapida/accesso rapido.

MAT-2.2.3 (K3) Preparare i test per valutare l'impatto su un'app delle impostazioni preferite dell'utente tra quelle fornite da un sistema operativo.

HO-2.2.3 (H3) Testare un'app in esecuzione modificando le opzioni del valore di input per le preferenze fornite dal sistema operativo.

MAT-2.2.4 (K2) Distinguere tra diversi test richiesti per applicazioni native, web e ibride.

HO-2.2.4 (H0) (Opzionale) Identificare i test necessari per le app, a seconda del tipo di app.

MAT-2.2.5 (K1) Tenere presente i test richiesti per le app disponibili su più piattaforme o versioni del sistema operativo.

MAT-2.2.6 (K1) Tenere presente i test richiesti per la coesistenza e l'interoperabilità con altre app.

2.3 Testing dei Vari Metodi di Connettività

MAT-2.3.1 (K2) Riassumere i test per le verifiche della connettività, compresi quelli su reti, quando si utilizza il Bluetooth e quando si passa alla modalità aereo.

HO-2.3.1 (H0) (Opzionale) Eseguire test su un'applicazione che sta trasferendo dati al server quando il telefono passa dalla connettività Wi-Fi alla rete dati in base alla potenza del segnale disponibile.

2.1 Testing di Compatibilità con l'Hardware del Dispositivo

2.1.1 Testare le caratteristiche del dispositivo

Diversi tipi di dispositivi con funzionalità differenti si traducono in test di compatibilità che devono essere condotti su un gran numero di dispositivi. Per questo motivo, è necessario assegnare una priorità ai dispositivi target per i test. Per stabilire tale priorità si utilizzano i dati di mercato, come discusso nella sezione 1.1, per selezionare un portfolio di dispositivi più appropriato per il mercato di riferimento. La selezione del portfolio di dispositivi di solito è un compromesso tra copertura del mercato, costi e rischi.

Le applicazioni possono essere installate su diversi tipi di dispositivi con le seguenti funzionalità:

- Diversi metodi per lo spegnimento
- Diversi metodi di navigazione
- Uso di hard e soft keyboard
- Varie funzionalità hardware come:
  - Radio
  - USB

- Bluetooth
- Fotocamere
- Altoparlanti
- Microfono
- Ingresso per le cuffie

Nessuna di queste funzionalità dovrebbe interferire negativamente con le operazioni dell'applicazione.

Le funzionalità del dispositivo hanno molte varianti e possono differire anche tra i diversi modelli di dispositivo realizzati dallo stesso produttore. Sono comunemente utilizzati per differenziare i segmenti di mercato e possono cambiare rapidamente nel tempo. Ad esempio, al momento è abbastanza comune che i dispositivi di fascia alta e media abbiano sensori di impronte digitali, mentre i dispositivi di fascia bassa no. Ciò può cambiare col tempo. Alcuni anni fa, i sensori di impronte digitali non erano inclusi in alcun dispositivo mobile. A causa di questa variabilità, il tester necessita di una chiara comprensione dei dispositivi e delle funzionalità attese dai suoi utenti. Il tester deve creare il portfolio dispositivi e progettare i test corrispondenti di conseguenza.

In generale, non è sufficiente verificare se l'applicazione funziona correttamente con le funzionalità previste. È necessario verificare anche che l'app funzioni come previsto se una determinata funzionalità è assente. Ad esempio, un'app che supporta l'utilizzo della fotocamera anteriore e posteriore non dovrebbe arrestarsi in modo anomalo se installata ed eseguita su un dispositivo con più fotocamere, solo una fotocamera o nessuna fotocamera.

2.1.2 Testare display diversi

I display dei dispositivi possono avere diverse dimensioni dello schermo, dimensioni del viewport, fattori di forma (aspect ratio) e risoluzioni misurate in pixel per pollice (ppi) e punti per pollice (dpi). La frammentazione dei dispositivi rende necessario stabilire una priorità. Dovrebbero essere creati test che esercitino l'interfaccia utente su vari dispositivi con le dimensioni dello schermo, le risoluzioni e i fattori di forma più comuni del mercato di riferimento.

È necessario eseguire test su diversi display per verificare quanto segue:

- L'app ridimensiona tutti gli elementi dell'interfaccia utente in base alla densità e alle dimensioni attuali dello schermo.
- Gli elementi dell'interfaccia utente non si sovrappongono.
- Non si verificano problemi di usabilità o di touch.
- Non si verificano problemi nel restringimento delle immagini nel caso di elevato dpi/ppi.

2.1.3 Test sulla temperatura del dispositivo

A differenza dei computer desktop, i dispositivi mobili reagiscono in modo diverso all'aumento della temperatura del dispositivo.

I dispositivi mobili potrebbero surriscaldarsi per una serie di motivi come la carica della batteria, il carico di lavoro intenso, le app in esecuzione in background e l'utilizzo continuo della rete dati cellulare, del Wi-Fi o del GPS.

Il surriscaldamento può influire su un dispositivo in quanto il dispositivo cerca di ridurre il livello di riscaldamento e preservare i livelli della batteria. Ciò può includere una riduzione della frequenza della CPU, la liberazione della memoria e lo spegnimento di parti del sistema.

In questo caso, vi è anche un possibile impatto sulla funzionalità dell'app e ciò deve essere considerato durante la pianificazione dei test. I test devono essere progettati in modo da consumare molta energia, generando calore per un lungo periodo ininterrotto. Il software sotto test non deve quindi mostrare alcun comportamento imprevisto.

2.1.4 Test sui sensori di input del dispositivo

I dispositivi mobili ricevono vari di tipi di input dai sensori che utilizzano, ad esempio, GPS, accelerometri, giroscopi e magnetometri a 3 assi o che reagiscono a input di pressione, temperatura, umidità, battito cardiaco, luce o sono di tipo touchless.

I test sui diversi sensori di input del dispositivo verificano quanto segue:

- L'app funziona come previsto per ciascuno dei sensori disponibili. Ad esempio, l'app deve essere testata per vari tipi di movimento, come quello circolare e il movimento avanti e indietro (come nel camminare).
- Le funzionalità che reagiscono all'illuminazione esterna reagiscono correttamente in varie condizioni di illuminazione.
- Gli ingressi e le uscite audio rispondono correttamente in combinazione con i pulsanti soft e hard del volume, microfoni, altoparlanti wired e wireless e in varie condizioni di suono ambientale.
- La localizzazione della posizione è accurata nelle seguenti condizioni:
    - Attivando e disattivando il GPS.
    - Con diverse qualità del segnale GPS.
    - Laddove l'app necessiti di ricorrere a vari altri metodi di localizzazione, tra cui Wi-Fi, posizione della stazione radio base o inserimento manuale della posizione.

2.1.5 Test sui vari metodi di input

I test sui diversi metodi di input del dispositivo verificano quanto segue:

- Dal momento che i telefoni cellulari consentono l'installazione di varie soft keyboard, l'app è in grado di funzionare almeno con quelle fornite dai principali produttori di dispositivi e quelle più ampiamente utilizzate.
- L'app assicura che la tastiera si apra per impostazione predefinita con layout e tasti appropriati quando richiesto.
- Quando un utente posiziona una o più dita sul touch screen, l'applicazione interpreta tale pattern come un particolare gesture o comando. I gesture tipici includono press/touch, double touch, multi-touch, swipe, tap, double tap, drag e pinch open/close.

- Ogni schermata dell'app deve rispondere correttamente ai gesture o ad altri input in modo appropriato sulla base della schermata e ignorare tutti i gesture o input non supportati.
- Le fotocamere utilizzate dalle app sono in grado di acquisire immagini e video, scansionare codici a barre, codici QR e documenti e misurare le distanze.
- Laddove sia disponibile una fotocamera anteriore e una posteriore, quella appropriata per l'app si apre per impostazione predefinita. Ad esempio, quando una video chat richiede l'accensione della fotocamera anteriore per impostazione predefinita, le app devono essere testate per verificare i casi in cui esse usano l'input della fotocamera e quelli dove non la usano. Inoltre, i test devono garantire che il software sotto test funzioni correttamente se è presente solo una fotocamera (anteriore o posteriore) anziché due. Ciò è particolarmente rilevante se il software sotto test utilizza una fotocamera particolare e questa non è presente nel dispositivo.

2.1.6 Test sul cambio di orientamento dello schermo

I sensori di movimento vengono utilizzati per rilevare i cambiamenti di orientamento e attivare un passaggio tra le modalità orizzontale e verticale (e viceversa) con i relativi cambi di layout necessari nell'interfaccia utente.

I test eseguiti dopo un cambio di orientamento dello schermo verificano quanto segue:

- Usabilità e comportamento funzionali corretti quando viene eseguito il passaggio alla modalità verticale o orizzontale.
- L'app mantiene il suo stato.
- I campi dei dati di input conservano i dati già acquisiti.
- I campi dei dati di output mostrano gli stessi dati mantenendo la sessione corrente.

I test eseguiti dopo un cambio di orientamento dello schermo non dovrebbero concentrarsi solo su un singolo cambio poiché i problemi di rendering o di stato potrebbero non verificarsi sempre dopo un singolo cambio. I test dovrebbero quindi essere eseguiti con diversi cambi ininterrotti tra le modalità verticale e orizzontale.

Dovrebbero essere progettati test che cambiano orientamento più volte nei vari stati di un'interfaccia utente, con e senza dati. L'app dovrebbe comportarsi come previsto, mantenendo il suo stato senza alcuna perdita o modifica dei dati.

2.1.7 Test sugli interrupt comuni

Tipi comuni di interrupt del dispositivo includono chiamate vocali, messaggi, caricabatterie in funzione, memoria scarsa e altre notifiche. Gli interrupt avviati dall'utente derivano da azioni come passare da un'app all'altra durante l'utilizzo o impostare il dispositivo in standby mentre l'app è in esecuzione.

I test sugli interrupt verificano quanto segue:

- L'app gestisce correttamente tutti gli interrupt sopra menzionati senza impatto negativo sul comportamento dell'app.

- L'app continua a funzionare correttamente, preservando il suo stato, i dati e le sessioni indipendentemente dall'interrupt.
- Laddove il dispositivo abbia una modalità "do-not-disturb" di blocco delle notifiche, l'app deve assicurarsi che le varie condizioni vengano utilizzate correttamente. Questi test devono essere eseguiti anche quando la modalità "do-not-disturb" è disattivata dopo essere stata attiva per un lungo periodo di tempo. Ciò comporta la ricezione massiva di più notifiche.
- I test dovrebbero prevedere di ricevere interrupt durante l'utilizzo dell'app per assicurarsi che tali interrupt non abbiano un impatto negativo. Ad esempio, rispondere a una telefonata durante l'utilizzo dell'app deve poi consentire di riportare l'utente allo stato in cui si trovava al momento dell'interrupt.

2.1.8 Test sulle autorizzazioni di accesso alle funzionalità del dispositivo

Le app devono accedere a varie cartelle come contatti e immagini e a sensori come fotocamera e microfono. Negare l'accesso durante l'installazione o modificarlo in un secondo momento potrebbe influire sul comportamento dell'app.

I test sulle autorizzazioni di accesso verificano quanto segue:

- L'app è in grado di funzionare con autorizzazioni ridotte; essa chiede all'utente di concedere queste autorizzazioni e non si chiude in modo inspiegabile.
- Le autorizzazioni sono richieste solo per le risorse rilevanti per la funzionalità dell'app; non sono consentite autorizzazioni generali per risorse non correlate.
- La funzionalità dell'app risponde correttamente se un'autorizzazione viene revocata o rifiutata durante l'installazione.
- Qualsiasi richiesta di autorizzazione rilasciata dall'app è corretta e giustificata.

Per verificare le autorizzazioni di accesso, un tester deve sapere perché l'app ha bisogno di ciascuna autorizzazione e in che modo la funzionalità deve essere influenzata se l'autorizzazione viene revocata o rifiutata durante l'installazione. I test dovrebbero prevedere di rifiutare le autorizzazioni durante l'installazione e nel concederle a valle della stessa.

2.1.9 Test sui consumi e lo stato di carica

Il test sui consumi e sullo stato di carica verifica quanto segue:

- Stato di carica della batteria e difetti relativi al consumo.
- Integrità dei dati in condizioni di basso consumo e batteria scarica.
- Consumo energetico mentre l'app è attiva e in condizioni di utilizzo intenso e scarso.
- Consumo energetico mentre l'app è in background.

Questi test devono essere pianificati attentamente poiché devono essere eseguiti ininterrottamente per un lungo periodo di tempo. Ad esempio, potrebbe essere necessario lasciare il dispositivo incustodito con l'app in background o in foreground, ma senza che il dispositivo sia utilizzato. Strumenti come gli analizzatori di log sono necessari per estrarre informazioni sugli andamenti di consumo della batteria.

2.2 Testing delle Interazioni tra App e Software del Dispositivo

2.2.1 Test sulle notifiche

Esistono vari meccanismi utilizzati dal sistema operativo per visualizzare le notifiche. A volte il sistema operativo può ritardare la visualizzazione delle notifiche o non visualizzarle affatto nel tentativo di ottimizzare il consumo di energia. Devono essere considerate le seguenti condizioni di test:

- La corretta gestione delle notifiche ricevute quando l'app è in foreground o background, soprattutto in condizioni di batteria scarica.
- Se le notifiche consentono l'interazione diretta con il contenuto dell'app (ovvero, senza aprire l'app stessa), l'interazione dell'utente deve essere fornita dall'app in un secondo momento. Se, ad esempio, l'utente risponde a una notifica, deve essere possibile accedere a tale risposta dall'interno dell'app in un secondo momento.
- Se le notifiche consentono l'accesso all'app, è necessario aprire la pagina corrispondente dell'app anziché la schermata principale quando la notifica contiene un collegamento diretto (deep link) a tale pagina.

2.2.2 Test sui collegamenti ad accesso rapido

I collegamenti ad accesso rapido (quick access link) come le scorciatoie (shortcut) delle app in Android e Force-touch o 3d-touch per iOS possono essere forniti dal software sotto test. Queste funzionalità eseguono un sottoinsieme delle funzionalità dell'applicazione dalla schermata principale senza avviare effettivamente l'intera app.

Devono essere considerate le seguenti condizioni di test:

- Laddove alcune funzionalità siano disponibili solo su una particolare versione del sistema operativo, il sistema sotto test deve comportarsi correttamente se installato su versioni del sistema operativo che offrono o non offrono tali funzionalità.
- Le azioni eseguite nei collegamenti ad accesso rapido si riflettono correttamente nell'app quando vengono aperte.

2.2.3 Test sulle preferenze dell'utente fornite dal sistema operativo

Eventuali preferenze (impostazioni) fornite agli utenti dal sistema operativo devono essere testate. Se una determinata impostazione delle preferenze non viene rispettata dall'app, si creano le condizioni per un'esperienza utente negativa. Ad esempio, se il dispositivo è in modalità silenziosa, l'app non dovrebbe riprodurre suoni.

Devono essere considerate le seguenti condizioni di test:

- Gli utenti possono modificare le opzioni delle preferenze più comuni come suono, luminosità, rete, modalità di risparmio energetico, data e ora, fuso orario, lingue, tipo di accesso e notifiche.
- Le app seguono le preferenze impostate comportandosi di conseguenza.

2.2.4 Test sui diversi tipi di app

Si possono eseguire test specifici in base al tipo di app mobile (vedere la sezione 1.4). Devono essere considerate le seguenti condizioni di test:

- Per le app native:
    - Compatibilità del dispositivo

- Utilizzo delle funzionalità del dispositivo

- Per le app ibride:
  - Interazione dell'app con le funzionalità native del dispositivo
  - Potenziali problemi di prestazioni dovuti al livello di astrazione
  - Usabilità (look and feel) rispetto alle app native sulla piattaforma in questione
- Per le app web:
  - Test per determinare la compatibilità cross-browser dell'app con vari browser mobili comuni
  - La funzionalità non è influenzata da vari motori JavaScript
  - Utilizzo delle funzionalità del sistema operativo (ad esempio, selezione data e apertura della tastiera appropriata)
  - Usabilità (look and feel) rispetto alle app native sulla piattaforma in questione

2.2.5 Test per l'interoperabilità con più piattaforme e versioni del sistema operativo

Le società di software supportano spesso app su più sistemi operativi. Ogni sistema operativo mobile ha i propri limiti che devono essere presi in considerazione durante il test delle app. I tester devono essere consapevoli delle specifiche di ciascuna piattaforma testata per assicurarsi che l'app funzioni come previsto, pur rimanendo conforme al “look and feel” della piattaforma.

Devono essere considerate le seguenti condizioni di test:

- Gestione di interrupt, notifiche e ottimizzazioni (ad es. per il risparmio energetico).
- Funzionalità corretta delle applicazioni che girano su più piattaforme sia che tali applicazioni condividano parte del codice che qualora siano state create utilizzando framework di sviluppo per più piattaforme. Si noti che se le applicazioni non condividono il codice, è come testare due diverse applicazioni ed è necessario testare tutto.
- Test sulla compatibilità all’indietro qualora una piattaforma utilizzi diverse versioni del sistema operativo.
- Test su funzionalità nuove o modificate implementate sulle piattaforme. Ad esempio, in Android l'introduzione del framework Doze ha richiesto test sia sulle varie versioni del sistema operativo che supportano questo framework che su quelle che non lo supportano.

2.2.6 Test sull'interoperabilità e coesistenza con altre app sul dispositivo

È abbastanza comune per le app interagire tra loro quando sono installate su un dispositivo. Esempi tipici sono le app per i registri dei contatti e le app di posta elettronica.

Devono essere considerate le seguenti condizioni di test:

- Il trasferimento dei dati tra il sistema sotto test e l'app utilizzata è corretto.
- Non vengono causati danni ai dati dell'utente archiviati all'interno di un'app utilizzata.
- Comportamenti conflittuali. Ad esempio, un'app potrebbe disattivare il GPS per risparmiare energia, mentre un'altra app potrebbe attivarlo automaticamente.

Con milioni di app sul mercato, è realisticamente impossibile testarle tutte per garantirne la coesistenza. Tuttavia, tali potenziali problemi dovrebbero essere considerati e testati in base al loro rischio.

2.3 Testing dei Vari Metodi di Connettività

I dispositivi mobili possono utilizzare vari metodi per connettersi alle reti (vedere la sezione 1.5). Questi includono reti cellulari come 2G, 3G, 4G e 5G, nonché Wi-Fi e altri tipi di connessione wireless come NFC o Bluetooth.

Bisogna considerare le seguenti alternative quando si eseguono test sulla connettività:

- Gli emulatori/simulatori di dispositivi possono simulare varie connessioni di rete e alcuni fornitori di servizi di accesso ai dispositivi remoti li includono tra le loro funzionalità. Gli emulatori/simulatori hanno tuttavia un valore limitato.
- Configurare la propria rete mobile per supportare vari tipi di connessione e quindi applicare la manipolazione della larghezza di banda. Questa è un'alternativa molto costosa.
- I field test sono potenzialmente un'alternativa più economica, ma sono limitati per quanto riguarda la riproducibilità dei test.

Nell'uso nel mondo reale, i metodi di connettività sono diversi. Gli utenti possono essere connessi in modo continuo utilizzando una modalità particolare oppure passare da una modalità all'altra, ad esempio da Wi-Fi a rete cellulare (ad esempio, quando un utente esce di casa mentre utilizza l'app). L'utente può passare tra varie reti e versioni Wi-Fi/cellulari, nonché tra celle GSM. Mentre è in movimento, l'utente può persino trovarsi in punti morti (dead spot) senza alcuna rete. Inoltre, egli può disconnettersi deliberatamente, ad esempio, passando alla modalità aereo.

I test sulla connettività devono garantire che vengano prese in considerazione le seguenti condizioni di test:

- Funzionalità dell'app corretta con diverse modalità di connettività.
- Il passaggio da una modalità all'altra non provoca alcun comportamento imprevisto o perdita di dati.
- Vengono fornite all'utente informazioni chiare se la funzionalità è limitata a causa di una connessione di rete limitata o assente o se la larghezza di banda è bassa. Il messaggio dovrebbe indicare le limitazioni e le loro cause.

3. Tipi di Test Comuni e Processo di Test per le Applicazioni Mobili - 200 minuti

Parole chiave

terminazione anomala, accessibilità, code injection, testing esplorativo, field testing, euristica, installabilità, efficienza delle prestazioni, testing delle prestazioni, testing post-rilascio, testing di sicurezza, session-based test management, testing di stress, livello di test, processo di test, piramide di test, tour, laboratorio di usabilità, testing di usabilità

Obiettivi di Apprendimento per Tipi di Test Comuni e Processo di Test per le Applicazioni Mobili

3.1 Tipi di Test Comuni per le Applicazioni Mobili

MAT-3.1.1 (K3) Preparare test d'installabilità per le app mobili.
MAT-3.1.2 (K3) Preparare test di stress per le app mobili.
MAT-3.1.3 (K2) Fornire esempi di problemi di sicurezza relativi alle app mobili.
MAT-3.1.4 (K1) Ricordare considerazioni sul comportamento temporale e delle risorse per le app mobili.
MAT-3.1.5 (K3) Preparare test di usabilità per le app mobili.
HO-3.1.5 (H2) Scegliere un tour, una mnemonica o un'euristica, per testare l'usabilità di un'app usando il session-based test management.
Nota: HO-3.1.5 e HO-3.3.1, HO-3.3.2 e HO-3.3.3 possono essere combinati.
MAT-3.1.6 (K1) Riconoscere i tipi di test richiesti per testare i database delle app mobili.
MAT-3.1.7 (K2) Riassumere i test richiesti per l'internazionalizzazione (globalizzazione) e i test di localizzazione delle app mobili.
MAT-3.1.8 (K2) Riassumere la necessità di test di accessibilità nei test delle applicazioni mobili.

3.2 Livelli di Test Aggiuntivi per le Applicazioni Mobili

MAT-3.2.1 (K2) Descrivere i livelli di test aggiuntivi, come i field test e le attività extra associate necessarie per test efficaci delle applicazioni mobili.
MAT-3.2.2 (K2) Descrivere i test richiesti per richiedere l'approvazione dell'application store nell'ambito della pubblicazione dell'app.

3.3 Tecniche di Test Basate sull'Esperienza

MAT-3.3.1 (K1) Ricordare il session-based test management, le "personas" e le mnemoniche nel contesto dei test esplorativi mobili.
HO-3.3.1 (H2) Scegliere una mnemonica (o parte di essa) specifica del mobile application testing per il testing di un'app tramite il session-based test management.
Nota: HO-3.1.5 e HO-3.3.1, HO-3.3.2 e HO-3.3.3 possono essere eseguiti insieme.
MAT-3.3.2 (K2) Descrivere l'uso di tour ed euristiche come tecniche esplorative per i test delle applicazioni mobili.
HO-3.3.2 (H2) Scegliere un'euristica specifica del mobile per testare un'applicazione mobile.
Nota: HO-3.1.5 e HO-3.3.1, HO-3.3.2 e HO-3.3.3 possono essere combinati.

MAT-3.3.3 (K3) Utilizzare un tour specifico per dispositivi mobili (come il tour Feature) per testare un'app mobile.
HO-3.3.3 (H2) Scegliere un tour specifico del mobile per testare un'applicazione mobile. Nota: HO-3.1.5 e HO-3.3.1, HO-3.3.2 e HO-3.3.3 possono essere combinati.

3.4 Processo e Approcci di Test per Applicazioni Mobili
MAT-3.4.1 (K2) Abbinare il processo di test, come descritto in [ISTQB_CTFL_2018], alle esigenze del mobile application testing.
MAT-3.4.2 (K2) Descrivere gli approcci di test, ad ogni livello di test, specifici per il mobile application testing.

3.1 Tipi di Test Comuni per le Applicazioni Mobili

3.1.1 Test di installabilità

I tester devono concentrarsi sull'installazione, l'aggiornamento e la disinstallazione dell'app utilizzando i seguenti approcci:

- Application store

  Il processo di installazione può variare a seconda degli utenti dell'app. Gli utenti possono installare l'app dagli store marketplace come Google Play Store o Apple App Store. Gli utenti delle app aziendali dovranno eseguire test di installazione tramite un collegamento o un servizio di distribuzione come HockeyApp o App Center.

- Sideloading (copia e installazione di app)

  Alcuni sistemi operativi offrono la possibilità di installare l'applicazione copiandola su un dispositivo mobile e installandola dal file.

- Applicazioni desktop

  Sono disponibili applicazioni desktop come Apple iTunes (per iOS) o Android App Installer per l'installazione di app sullo smartphone. Il tester deve scaricare l'app in questa applicazione e utilizzare un cavo per installarla da lì sullo smartphone. La maggior parte di queste applicazioni desktop consente anche la disinstallazione dell'app.

L'installazione può essere eseguita con i seguenti metodi:

- OTA (Over-the-Air) via Wi-Fi o rete dati
- Cavo dati

Alcune delle condizioni di test che possono essere considerate includono:

- Installazione, disinstallazione e aggiornamento su memoria interna ed esterna (se supportata).
- Reinstallazione dell'app quando è stata scelta l'opzione "conserva i dati dell'app" durante la precedente disinstallazione.
- Reinstallazione dell'app quando l'opzione "conserva i dati dell'app" non è stata scelta durante la precedente disinstallazione.
- Annullare o interrompere l'installazione o la disinstallazione, ad esempio, spegnendo il dispositivo mobile durante il processo o disconnettendosi da Internet.

- Riprendere l'installazione interrotta, la disinstallazione e l'aggiornamento dopo l'annullamento o l'interruzione.
- Test relativi alle autorizzazioni. Ad esempio, alcune app richiedono l'autorizzazione per utilizzare la rubrica. Questo importante test deve verificare il comportamento dell'app qualora l'utente neghi l'autorizzazione. Ad esempio, in questo caso viene inviato un messaggio corrispondente che sia significativo per l'utente?
- Aggiornare l'app e verificare che non vengano persi dati.

Alcune app richiedono dispositivi jailbreaked (iOS) o rooted (Android) che forniscono all'utente i diritti amministrativi sul dispositivo. La maggior parte dei provider di piattaforme non supporta il jailbreaking/ rooting poiché potrebbe avere conseguenze legali. Potrebbe non essere necessario testare un'app che non richiede il jailbreaking/rooting per dispositivi jailbreaked/rooted.

3.1.2 Test di stress

Il testing di stress si concentra sulla determinazione dell'efficienza prestazionale (performance efficiency) dell'applicazione quando soggetta a condizioni oltre il carico normale. Lo stress testing in questo contesto è mirato solo al dispositivo mobile. I test di stress a livello di back-end sono descritti nel syllabus ISTQB® Performance Testing ([ISTQB_CTFL_PT_2018]) e ulteriori informazioni possono essere trovate lì se necessario.

Alcune delle condizioni di test che possono essere considerate per i test di stress includono:

- elevato utilizzo della CPU
- memoria insufficiente
- spazio su disco insufficiente
- stress della batteria
- fallimenti
- scarsa larghezza di banda
- numero molto elevato di interazioni dell'utente (potrebbe essere necessario simulare le condizioni della rete nel mondo reale)

Alcune di queste condizioni di stress possono essere create utilizzando strumenti come Monkey. Si tratta di uno strumento a linea di comando che viene eseguito dalla linea di comando della shell ADB [URL3] o, se possibile, manualmente, ad esempio utilizzando file di grandi dimensioni o altre app con elevato utilizzo della CPU o consumo di memoria.

3.1.3 Test di sicurezza

Poiché i test di sicurezza sono un argomento complesso, ISTQB® ha dedicato un syllabus dettagliato separato su questo argomento [ISTQB_CTAL_SEC_2016]. I principali problemi di sicurezza per le app mobili includono:

- Accesso a dati sensibili sul dispositivo.
- Trasferimento di informazioni unencrypted o archiviazione non sicura.

Alcune delle condizioni di test che possono essere considerate per i test di sicurezza includono:

- Test degli input per code injection e overflow.

- Encryption dei dati trasferiti.
- Encryption dei dati memorizzati localmente.
- Cancellazione di dati temporanei dopo l'uso o dopo una terminazione anomala.
- Cancellazione del testo nei campi della password.

Dovrebbero essere esaminate anche le prime 10 vulnerabilità relative ai dispositivi mobili descritte nel Open Web Application Security Project (OWASP) [URL2].

3.1.4 Test delle prestazioni (Performance Testing)

Se l'utente installa l'app ed essa non risponde abbastanza velocemente (ad esempio, in massimo 3 secondi), l'utente potrebbe disinstallarla a favore di un'app alternativa. Gli aspetti relativi al tempo e al consumo di risorse sono importanti fattori di successo per un'app e i test delle prestazioni vengono condotti per misurare questi aspetti.

L'efficienza delle prestazioni deve essere testata sul dispositivo stesso oltre all'interazione con il sistema di back-end e altri dispositivi mobili.

Il testing delle prestazioni dell'intero sistema deve essere eseguito come definito nella strategia di test e non è specifico per i dispositivi mobili. Consultare il syllabus dedicato ISTQB® sul Performance Testing [ISTQB_CTFL_PT_2018] per ulteriori dettagli.

Il testing delle prestazioni dell'app stessa dovrebbe contenere la cronometria per i workflow più importanti. Alcuni esempi per i workflow di un'app di online banking sono: "Login", "Cambia indirizzo" o "Bonifico bancario con PIN e TAN". Il tester dovrebbe quindi confrontare questa cronometria con app simili.

Oltre alle misure cronometriche è importante considerare le prestazioni percepite dall'utente. L'esperienza dell'utente può avere un impatto enorme sul tempo per cui egli è disposto ad attendere il completamento di una determinata funzione.

3.1.5 Test di usabilità

L'usabilità è molto importante per le app mobili poiché le statistiche dimostrano che un numero elevato di utenti disinstalla le proprie app entro pochi minuti dall'installazione a causa della scarsa usabilità o di scarse prestazioni, vedere [URL4].

Per questo motivo, si raccomanda che il design dell'esperienza utente (UX) consideri il “look and feel” della piattaforma su cui l'app deve essere utilizzata. Se la UX non è conforme alle aspettative dell'utente per la piattaforma scelta, ciò può avere un forte impatto negativo. Pertanto, un tester dovrebbe essere consapevole del “look and feel” dell'app sulla piattaforma utilizzata.

I test di usabilità possono essere condotti da un tester utilizzando vari tour di test ed euristiche disponibili. Considerare diverse “personas” è anche un supporto utile per i test di usabilità. Se necessario, per questo scopo si può usare un laboratorio di usabilità.

Nei progetti, le rilevazioni effettuate durante il test di usabilità sono principalmente solo osservazioni e non difetti. Il tester deve essere in grado di descrivere tali rilevazioni al team, al product owner o stakeholder analoghi. Per ottenere un'usabilità soddisfacente, un'app dovrebbe:

- spiegarsi da sé ed essere intuitiva.
- consentire errori dell'utente.
- essere coerente nell’utilizzo di termini/espressioni e nel comportamento.

- rispettare le linee guida di progettazione delle piattaforme.
- rendere le informazioni necessarie visibili e raggiungibili in ogni dimensione e tipo di schermo.

Consultare il syllabus specialistico ISTQB® sul testing di usabilità [ISTQB_FLUT_2018] per ulteriori dettagli.

3.1.6 Test del database

Molte app devono archiviare i dati localmente utilizzando vari meccanismi di archiviazione dei dati come file flat o database. Alcune delle condizioni di test da considerare per il test del database delle app mobili includono:

- Validazione dei problemi di archiviazione dei dati:
  - Sincronizzazione
  - Conflitti di caricamento
  - Sicurezza dei dati
  - Vincoli sui dati
  - Funzionalità CRUD (Create, Read, Update, Delete)
  - Ricerca
- Test di integrazione dei dati per i dati forniti dal dispositivo (ad es. contatti) o da app di terze parti (ad es. immagini, video e messaggi).
- Prestazioni di archiviazione dei dati sul dispositivo.

3.1.7 Test di globalizzazione e localizzazione

Il testing di internazionalizzazione (I18N)/globalizzazione dell'applicazione consiste nel testare un'app rispetto a posizioni, formati di date, numeri e valute differenti e rispetto alla sostituzione di stringhe effettive con pseudo-stringhe.

Il testing di localizzazione (L10N) include il testing di un'app con stringhe, immagini e workflow locali di una particolare regione. Ad esempio, le parole russe e tedesche potrebbero essere molto più lunghe di quelle in altre lingue. Poiché i dispositivi mobili hanno dimensioni e risoluzioni dello schermo diverse, dimensioni dello schermo limitate possono causare problemi con le stringhe tradotte. Questi problemi dovrebbero essere verificati mediante test standard di globalizzazione/localizzazione.

Un aspetto molto importante da verificare è il formato della data utilizzato, ad esempio ANNO - MESE - GIORNO o GIORNO - MESE - ANNO.

3.1.8 Test di accessibilità

Il testing di accessibilità viene eseguito per determinare la facilità con cui gli utenti con disabilità possono utilizzare un componente o un sistema. Per le app mobili, ciò può essere fatto usando le impostazioni di accessibilità del dispositivo e testando l'app per ogni impostazione.

Le linee guida per l'accessibilità sono disponibili presso i rivenditori della piattaforma e dovrebbero essere utilizzate. Ad esempio, sia Google [URL5] che Apple [URL6] hanno pubblicato linee guida per l'accessibilità per le rispettive piattaforme. Anche ricevere feedback dalle persone che necessitano di caratteristiche di accessibilità è utile.

Per il web mobile è stata pubblicata una guida sull'accessibilità dal W3C, che dovrebbe essere presa in considerazione [URL7].

3.2 Livelli di Test Aggiuntivi per le Applicazioni Mobili

Oltre ai consueti livelli di test dal componente fino ai test di accettazione descritti in [ISTQB_CTFL_2018], sono necessari anche livelli di test aggiuntivi per il mobile application testing.

3.2.1 Field test

Alcune applicazioni mobili richiedono field test (sul campo) per garantirne il corretto funzionamento nello scenario di utilizzo previsto degli utenti reali. Ciò potrebbe includere test su varie reti e su diversi tipi di tecnologie di comunicazione come Wi-Fi o la rete dati cellulare.

I field test dovrebbero includere l'uso e le commutazioni tra differenti stazioni radio base, reti dati cellulari e Wi-Fi durante l'utilizzo dell'app. Dovrebbero essere eseguiti test con velocità di download e intensità del segnale variabili e test che includono la gestione di punti ciechi (blind spot).

I field test richiedono un'attenta pianificazione e l'identificazione di tutti gli elementi necessari per eseguirli, come tipi di dispositivi appropriati, le reti Wi-Fi, i piani delle reti dati cellulari sui diversi carrier e l'accesso a vari modi di trasporto necessari per fornire una copertura adeguata. Inoltre, si devono programmare le rotte, le modalità di trasporto e il momento del giorno in cui i test devono essere eseguiti.

L'usabilità di un'app è un altro aspetto importante da includere durante l'esecuzione di field test. I test dovrebbero comprendere fattori ambientali quali la temperatura e condizioni simili correlate allo scenario di utilizzo.

3.2.2 Test per l'approvazione degli store di app e test post-release

Prima di inviare un'app per la pubblicazione, è necessario che essa superi con successo alcuni test basati su checklist per garantire l'approvazione da parte degli store di app. Se la versione di rilascio è relativa ad un aggiornamento, è necessario eseguire anche i test relativi a tale aggiornamento.

Le checklist si basano in genere su linee guida, come quelle specifiche per i sistemi operativi, per la progettazione dell'interfaccia utente e per l'utilizzo delle librerie e delle API fornite dagli application store.

Il processo di approvazione potrebbe richiedere del tempo dopo la richiesta. Se vengono rilevati problemi durante il processo di approvazione, potrebbe essere necessario inviare una nuova versione che richiederà tempo aggiuntivo. Questa situazione richiede un'attenta valutazione durante la pianificazione e il testing del progetto.

Un ulteriore livello di test è quello dei test "post-release". I test a questo livello includono il download e l'installazione dell'applicazione dagli application store.

3.3 Tecniche di Test Basate sull'Esperienza

3.3.1 “Personas” e mnemoniche

Le “personas” sono personalità inventate che rappresentano clienti reali. Esse hanno motivazioni, aspettative, problemi, abitudini e obiettivi propri e sono utili quando è necessario imitare il comportamento reale dell'utente.

Una “persona” potrebbe avere nome, sesso, età, reddito, background educativo e posizione geografica. In un contesto mobile, le “personas” possono usare altre app, controllare il proprio dispositivo mobile x volte all'ora e possono avere altri dispositivi e tratti personali.

Una mnemonica è un aiuto per ricordare qualcosa. Nel contesto del testing, ogni lettera in una mnemonica rappresenta una tecnica, un metodo di testing o una serie elementi di attenzione per il testing. Un esempio di mnemonica è SFiDPOT [URL8]. Le lettere in tale mnemonica hanno i seguenti significati:

S (Structure) - Struttura (ad esempio, elementi dell'interfaccia utente, altri elementi dell'applicazione e il loro ordine e rapporto gerarchico)

F (Function) - Funzione (ad esempio, le funzionalità desiderate sono funzionanti, disponibili e rispettano i requisiti, ecc.)

i (input) - Input (ad esempio, tutti gli input richiesti come gli input da tastiera, i sensori e la fotocamera sono disponibili ed elaborati correttamente)

D (Data) - Dati (ad esempio, i dati vengono memorizzati (anche su scheda SD), modificati, aggiunti ed eliminati come definito nei requisiti)

P (Platform) - Piattaforma (ad esempio, le funzioni specifiche del sistema operativo sono disponibili in base alle impostazioni del dispositivo; la piattaforma include lo store per il download dell'app)

O - Operations (ad esempio, sono disponibili le attività dell'utente comune, come lo spostamento tra reti di telefonia mobile e Wi-Fi)

T (Time) - Tempo (ad esempio, la gestione e visualizzazione di fusi orari, ora e date)

Una mnemonica ed euristica che si occupa specificamente di dispositivi mobili è I SLICED UP FUN [URL9]. Le lettere in essa contenute hanno i seguenti significati:

I – Input

S – Store

L – Location (Posizione)

I – Interactions e interrupts (Interazioni e interrupt)

C – Communication (Comunicazione)

E – Ergonomics (Ergonomia)

D – Data (Dati)

U – Usability (Usabilità)

P – Platform (Piattaforma)

F – Function (Funzione)

U – User scenarios (Scenari utente)

N – Network (Rete)

3.3.2 Euristiche

Un approccio euristico è un approccio alla risoluzione dei problemi, all'apprendimento e alla scoperta che utilizza un metodo pratico basato su "regole empiriche". Tale approccio non è ottimale o perfetto, ma può essere considerato sufficiente per raggiungere gli obiettivi nell'immediato.

Esistono molte euristiche per i test delle applicazioni mobili. La maggior parte delle mnemoniche possono essere usate come euristiche, ma non tutte le euristiche sono mnemoniche.

3.3.3 Tour

I tour vengono utilizzati nel testing esplorativo per consentire l'esplorazione di un'applicazione da punti di vista e focus specifici. I tour possono essere effettuati per capire come funziona un'applicazione e per creare modelli per il workflow. Essi forniscono inoltre un metodo efficace per i field test.

Un esempio di tour è quello del “Landmark” in cui un utente imita le visite di un turista in una città visitando le attrazioni principali. La tabella seguente mostra come le visite effettuate durante il tour possano essere utilizzate in analogia ai passaggi da seguire nei test delle applicazioni mobili.

Visite nel tour Landmark	Analogia per i test delle applicazioni mobili
Il quartiere storico	Codice legacy
Zona commerciale L’ora di punta	Business logic dell'app Avvio e arresto dell'app
Il quartiere turistico	Parte dell'app utilizzata dai nuovi utenti
Il quartiere dell'hotel	Parti dell'app attive solo in modalità sleep

La combinazione di session-based test management (vedere la sezione 3.3.4) e tour, incluso l'uso di euristiche e mnemoniche, aiuta a migliorare l'efficacia del mobile application testing.

La tabella seguente mostra alcuni buoni esempi di tour per i test delle app e le aree che coprono per fornire idee per i test. Alcuni di questi si trovano in [Kohl17].

Tour per test di app	Aspetto coperto
Supermodel	“Look and feel” e usabilità
Landmark	Le funzionalità più importanti nell'app
Sabotage	Robustezza
Feature	Nuove funzionalità
Scenario	Intero flusso di lavoro nell'app in combinazione con le user story
Connectivity	Connettività utilizzata, come Wi-Fi, GSM
Location	Lingua, date e numeri corretti
Light	Visibilità in diverse condizioni di illuminazione, come buio, esterno, luce rossa.
Low battery	Perdite di dati nell'app causate da bassi livelli di energia.
Ulteriori tour per test di app	Aspetto coperto in [Kohl17]
Gesture	Usare tutti i gesture ove possibile
Orientation	Cambiare l’orientamento
Change your mind	Tornare indietro

Motion	Eseguire vari tipi di movimenti
Location	Muoversi
Connectivity	Cambiare i tipi di connessione o le posizioni spostandosi
Comparison	Confronto con altri tipi di dispositivi
Consistency	Controllo della coerenza delle schermate, della GUI

3.3.4 Session-Based Test Management (SBTM)

Il Session-Based Test Management (SBTM) consente di gestire i test esplorativi in periodi di tempo di determinata durata (timebox). Una sessione è composta da tre attività:

- Impostazione (setup) della sessione
- Progettazione ed esecuzione dei test
- Analisi e reporting delle problematiche rilevate

Il SBTM utilizza in genere un foglio di sessione (session sheet) contenente un test charter che fornisce gli obiettivi della sessione di test. Inoltre, il foglio di sessione viene utilizzato per documentare le attività di esecuzione dei test svolte.

Il testing esplorativo è una tecnica di test basata sull'esperienza che può costituire un approccio efficace per testare le applicazioni mobili. Le tecniche di test basate sull'esperienza sono descritte in [ISTQB_CTFL_2018].

3.4 Processo e Approcci di Test per Applicazioni Mobili

3.4.1 Processo di test

Le principali attività del processo di test ISTQB® sono descritte in [ISTQB_CTFL_2018] e sono applicabili anche ai test delle applicazioni mobili.

Vi sono altri aspetti specifici dei test delle applicazioni mobili che devono essere sempre considerati come parte del processo di test ISTQB®.

Principale gruppo di attività nel processo di test	Aree comuni da considerare per i test delle applicazioni mobili	Riferimenti al syllabus
Pianificazione dei test	• Combinazioni di dispositivi che devono essere testati. • Uso di emulatori e simulatori mobili come parte dell'ambiente di test. • Sfide speciali nei test delle app mobili. • Tipi di test richiesti specificatamente per i test delle applicazioni mobili.	• Sezione 4.3 • Sezione 1.7 • Sezione 3.2
Analisi e progettazione	• Test di approvazione degli app store. • Field test. • Compatibilità del dispositivo. • Tipo di laboratori da utilizzare. • Tipi di test richiesti specificatamente per i test delle applicazioni mobili.	• Sezione 3.2.2 • Sezione 3.2.1 • Sezione 3.2

Implementazione ed esecuzione del test	• Field test. • Test post-release di download e installabilità. • Tecniche di test basate sull'esperienza	• Sezione 3.2.1 • Sezione 3.1.1 [ISTQB_CTFL_2018]
Implementazione ed esecuzione del test	• I test si basano su linee guida della piattaforma per l'interfaccia utente e per gli app store. • I test basati su linee guida verranno generalmente eseguiti dai fornitori della piattaforma per il loro processo di approvazione dell'application store • Si consiglia di eseguirli a livello di provider dell'applicazione, prima di consegnare l'app ai provider della piattaforma al fine di evitare il possibile rifiuto.	

3.4.2 Approcci al test

Il mobile application testing include attività che devono essere eseguite dagli sviluppatori e dai tester.

Determinare la giusta profondità dei test per ogni loro livello (ad esempio test dei componenti, test di integrazione, test di sistema, field test, approvazione dell'application store, test post-release e test di accettazione utente) è importante per fornire prodotti di buona qualità. La profondità dei test necessaria per ogni livello di test dipende da molti fattori, come l'architettura delle app, la loro complessità e l'insieme di utenti previsto.

Le piattaforme di sviluppo mobili offrono una varietà di strumenti per supportare i test a vari livelli. Comprendere gli strumenti e come possono essere applicati a un determinato livello è molto importante. Ad esempio, un simulatore/emulatore mobile può essere utilizzato a livello di test dei componenti se è necessario sfruttare il framework fornito dalla piattaforma e le API di instrumentazione. Inoltre, i simulatori/emulatori mobili possono essere utilizzati a livello di test di sistema quando i dispositivi reali non sono disponibili. Ciò consente di verificare la funzionalità e alcuni aspetti di usabilità e dell'interfaccia utente.

Inoltre, l'implementazione anticipata può essere un punto chiave per garantire che i dispositivi siano impostati correttamente e che tutti i prerequisiti per l'esecuzione siano rispettati in tempo.

Anche i test di unità e quelli di integrazione sono importanti, così come i test manuali (specialmente nella fase di field test). È molto comune per le app mobili rovesciare la piramide del test (Test Pyramid) [Knott15]. Ciò significa che possono esserci molti test manuali.

4. Piattaforme, Strumenti e Ambiente delle Applicazioni Mobili - 80 minuti

Parole chiave

emulatore, field testing, test basati sulla prossimità, laboratorio di test remoto, simulatore

Obiettivi di Apprendimento per Piattaforme, Strumenti e Ambiente delle Applicazioni Mobili

4.1 Piattaforme di Sviluppo per le Applicazioni Mobili
MAT-4.1.1 (K1) Ricordare gli ambienti di sviluppo utilizzati per lo sviluppo di applicazioni mobili.

4.2 Strumenti Comuni della Piattaforma di Sviluppo
MAT-4.2.1 (K1) Ricordare alcuni degli strumenti comuni forniti come parte delle piattaforme di sviluppo delle applicazioni.
HO-4.2.1 (H1) Utilizzare gli strumenti del kit di sviluppo software per acquisire schermate, estrarre un log e simulare gli eventi in arrivo.

4.3 Emulatori e Simulatori
MAT-4.3.1 (K2) Comprendere le differenze tra emulatori e simulatori.
MAT-4.3.2 (K2) Descrivere l'uso di emulatori e simulatori per i test delle applicazioni mobili.
HO-4.3.2 (H1) Creare ed eseguire un dispositivo simulato/emulato, installare un'app ed eseguire alcuni test su di essa.

4.4 Allestire un Laboratorio di Test
MAT-4.4.1 (K2) Distinguere tra vari approcci per allestire un laboratorio di test.

4.1 Piattaforme di Sviluppo per le Applicazioni Mobili

Gli ambienti di sviluppo integrato (IDE) sono disponibili sul mercato per vari sviluppi di app mobili. Questi IDE dispongono di vari strumenti che aiutano a progettare, codificare, compilare, installare, disinstallare, monitorare, emulare, effettuare il logging e testare le app.

Ad esempio, Android Studio può essere utilizzato per lo sviluppo di app per Android mentre Xcode per lo sviluppo di app per iOS. Questi differiscono dai normali IDE per il supporto aggiuntivo che offrono per le piattaforme mobili.

Sono inoltre disponibili alcuni framework per più piattaforme (cross-platform) che supportano lo sviluppo di applicazioni mobili. Questi funzionano su più piattaforme e non richiedono la codifica specifica.

4.2 Strumenti Comuni della Piattaforma di Sviluppo

I kit di sviluppo software di solito forniscono varie utility che aiutano nello sviluppo e nel testing delle applicazioni. Queste utility offrono una vasta gamma di funzionalità, come catturare

schermate, estrarre log, inviare eventi casuali e notifiche al dispositivo, monitorare vari parametri come l'utilizzo della memoria e della CPU e creare dispositivi virtuali.

Alcuni esempi di tali strumenti sono Android Virtual Device (AVD) Manager, Android Debug Bridge (ADB) e Android Device Monitor per Android e Instruments per iOS.

4.3 Emulatori e Simulatori

4.3.1 Panoramica di emulatori e simulatori

Nel contesto di questo syllabus, i termini "emulatore" e "simulatore" si riferiscono all'emulatore mobile o al simulatore mobile. Tali termini sono talvolta usati in modo intercambiabile ma errato. Per le definizioni, consultare il glossario nel capitolo 8.

Un simulatore modella l'ambiente di runtime, mentre un emulatore modella l'hardware e utilizza lo stesso ambiente di runtime dell'hardware fisico. Le applicazioni testate su un simulatore sono compilate in una versione dedicata, che funziona nel simulatore ma non su un dispositivo reale. Pertanto, il simulatore è indipendente dal sistema operativo reale.

Al contrario, le applicazioni compilate per essere distribuite e testate su un emulatore vengono compilate nel byte-code effettivo che potrebbe essere utilizzato anche dal dispositivo reale.

I simulatori e gli emulatori sono molto utili nelle prime fasi di sviluppo in quanto si integrano in genere con gli ambienti di sviluppo e consentono una rapida implementazione, test e monitoraggio delle applicazioni.

I simulatori vengono talvolta utilizzati anche in sostituzione dei dispositivi reali durante i test. Tuttavia, ciò avviene in maniera più limitata rispetto a quando si usano gli emulatori, poiché l'applicazione testata su un simulatore differisce a livello di byte-code dall'applicazione che verrà distribuita.

Gli emulatori vengono inoltre utilizzati per ridurre il costo degli ambienti di test sostituendo dispositivi reali per alcuni dei test. Non possono sostituire completamente un dispositivo perché potrebbero comportarsi in modo diverso rispetto al dispositivo mobile che tentano di imitare. Inoltre, alcune funzioni, come il (multi)touch, l'accelerometro e altre, potrebbero non essere supportate. Ciò è in parte causato dalle limitazioni della piattaforma utilizzata per eseguire l'emulatore.

4.3.2 Utilizzo di emulatori e simulatori

L'uso di emulatori e simulatori per i test delle applicazioni mobili può essere utile per vari motivi.

Ogni ambiente di sviluppo del sistema operativo mobile viene generalmente fornito con un proprio emulatore e simulatore in bundle. Sono anche disponibili emulatori e simulatori di terze parti.

Un tester può utilizzare qualsiasi emulatore o simulatore adatto al proprio scopo. Per usare un emulatore o un simulatore è necessario avviarli, installare l'app necessaria su di essi e quindi testare l'app stessa come se fosse sul dispositivo reale.

Di solito emulatori e simulatori consentono l'impostazione di vari parametri di utilizzo. Queste impostazioni potrebbero includere l'emulazione della rete a varie velocità e con diversi livelli di intensità del segnale e di perdita di pacchetti, il cambio di orientamento dello schermo, la generazione di interrupt e l'impostazione di dati sulla posizione GPS. Alcune di queste impostazioni possono essere molto utili perché possono essere difficili o costose da replicare con dispositivi reali, come le posizioni GPS globali o l'intensità del segnale.

La connessione agli emulatori per fini d'installazione potrebbe richiedere l'uso di strumenti da linea di comando come Android Debug Bridge (ADB) per Android o potrebbe essere stabilita dall'interno dell'ambiente di sviluppo integrato come con Xcode o Android Studio.

4.4 Allestire un Laboratorio di Test

Si utilizzano i seguenti approcci per allestire un laboratorio di test di applicazioni mobili:

Laboratorio on-premise

Con un laboratorio on-premise, tutti i dispositivi, gli emulatori e i simulatori si trovano sul posto. La selezione del dispositivo può essere effettuata sulla base di vari fattori come la classificazione del dispositivo (come riscontrato in Google o altri strumenti di analisi), il tipo e le versioni del sistema operativo, le dimensioni e la densità dello schermo, la disponibilità e i costi, le funzioni speciali e l'importanza nel pubblico di destinazione.

I vantaggi di un laboratorio on-premise includono la disponibilità di dispositivi per test specifici basati sulla prossimità e aspetti specifici dei sensori come batteria, touch e maggiore sicurezza.

L'allestimento di questo tipo di laboratorio può richiedere budget elevati a seconda dei dispositivi da acquistare e gestire. Ulteriori sfide includono la disponibilità tempestiva e le difficoltà con i test in diverse località e ambienti.

Laboratorio di test remoto

Questi laboratori sono importanti e utili per i test quando dispositivi o reti non sono fisicamente disponibili on-premise. L'accesso remoto ai dispositivi (Remote Device Access, RDA) è possibile tramite una connessione di rete a vari dispositivi ospitati nel data center del provider. Ogni potenziale fornitore RDA deve essere valutato per la conformità ai requisiti, soprattutto di sicurezza.

Alcuni laboratori remoti offrono le seguenti funzionalità aggiuntive:

- Versioni dedicate di dispositivi fisici (ad es. laboratorio di dispositivi mobili Samsung).
- Dispositivi generici solo per un sistema operativo e una versione specifici.
- Bracci robotici per eseguire operazioni relative a touch e gesture.
- Connessioni di rete privata virtuale (VPN) per consentire l'accesso al dispositivo.
- Connessioni cellulari con vari provider di reti cellulari.
- Strumenti e servizi di automazione.

Alcuni dei fattori da tenere a mente quando si utilizzano i laboratori di test remoti includono la ridotta responsiveness dei dispositivi e opzioni limitate per l'interazione con dispositivi tramite multi-touch e gesture. La scelta laboratori di test remoti può essere conveniente per un utilizzo sporadico, mentre tende a diventare generalmente più costosa per utilizzi protratti per lunghi periodi di tempo che richiedono una vasta gamma di dispositivi.

Altri fattori includono la disponibilità della piattaforma su richiesta rispetto alla necessità di ottenere l'accesso ai dispositivi mancanti nel laboratorio locale e la scalabilità del laboratorio, il quale può crescere e ridursi man mano che il progetto evolve.

Gli scenari di test che includono sensori come NFC/Bluetooth o il consumo della batteria sono spesso difficili da testare nel cloud. Tuttavia, le diverse posizioni geografiche dei laboratori remoti possono supportare i test che richiedono connessioni di rete e GPS.

Un laboratorio di test può utilizzare un approccio o una combinazione dei due approcci, a seconda del tipo di test che deve essere eseguito.

5. Automazione dell'Esecuzione dei Test - 55 minuti

Parole chiave

Application programming interface (API), test device-based, test user-agent-based, test report

Obiettivi di Apprendimento per l'Automazione dell'Esecuzione dei Test

5.1 Approcci di Automazione
MAT-5.1.1 (K2) Distinguere tra approcci e framework di automazione comuni per il mobile application testing.

5.2 Metodi di Automazione
MAT-5.2.1 (K2) Descrivere vari metodi di automazione per testare le app mobili.

5.3 Valutazione degli Strumenti di Automazione
MAT-5.3.1 (K1) Ricordare i vari parametri da considerare durante la valutazione degli strumenti di automazione dei test delle applicazioni mobili.

5.4 Approcci per l'Allestimento di un Laboratorio per i Test Automatizzati
MAT-5.4.1 (K2) Distinguere tra gli approcci comuni all'allestimento di laboratori di test, considerando vantaggi e svantaggi rispetto all'automazione dei test.

5.1 Approcci di Automazione

Esistono vari approcci e framework di automazione che possono essere utilizzati nei test delle applicazioni mobili. La scelta dell'approccio sarà in parte determinata dal tipo di applicazione da testare.

Due approcci di automazione dei test comunemente utilizzati sono:

- Test user-agent-based
- Test device-based

Il test user-agent-based utilizza la stringa di identificativo user-agent inviata dal browser per imitare un browser specifico su un determinato dispositivo. Questo approccio può essere utilizzato per l'esecuzione di applicazioni web mobili. Il test device-based comporta invece l'esecuzione dell'applicazione sotto test direttamente sul dispositivo. Questo approccio può essere utilizzato per tutti i tipi di applicazioni mobili.

Il tipo di applicazione può anche determinare il framework di automazione del test adatto per l'applicazione in esame. Le applicazioni web mobile possono essere testate utilizzando i comuni strumenti di automazione delle applicazioni web sul desktop, mentre le app native potrebbero richiedere strumenti specifici. I fornitori di piattaforme possono anche fornire strumenti di automazione dedicati alla piattaforma.

Gli approcci di automazione utilizzati per le applicazioni convenzionali sono spesso applicabili anche alle applicazioni mobili. Questi includono l'acquisizione/riproduzione, test data-driven,

keyword-driven e behavior-driven, come descritto nel syllabus ISTQB® Foundation Level [ISTQB_CTFL_2018] e nel syllabus ISTQB® Advanced Level Specialist Test Automation Engineer [ISTQB_CTAL_TAE_2016].

Le funzionalità chiave che un framework di test di applicazioni mobili dovrebbe in genere includere sono:

- Identificazione degli oggetti
- Operazioni sugli oggetti
- Test report
- Application programming interface ed estendibilità delle funzionalità
- Documentazione adeguata
- Integrazioni con altri strumenti
- Indipendenza dalla pratica di sviluppo dei test

5.2 Metodi di Automazione

Per sviluppare test automatici, il tester deve comprendere il meccanismo di registrazione o creazione degli script di automazione e in che modo accedere e interagire con gli oggetti grafici dell'applicazione come pulsanti, list box e campi di input.

Esistono diversi metodi per identificare un oggetto grafico utilizzato per l'automazione dei test delle applicazioni mobili. Questi includono il riconoscimento delle immagini, dell'OCR/testo e degli oggetti (web o nativo, a seconda del tipo di app).

Un tester di applicazioni mobili non deve solo avere esperienza col rilevamento e l'identificazione grafica degli oggetti, ma anche capire quale metodo di identificazione degli oggetti sarà maggiormente in grado di consentire un'efficace esecuzione dei test su una grande varietà di dispositivi mobili, in parallelo e in modo continuo.

Le principali differenze tra i metodi di creazione degli script sono:

Oggetto del confronto	Identificazione dell'oggetto	Confronto di immagini/OCR
Affidabilità	Finché l'identificatore è costante, è possibile modificare il layout dello schermo. Il rischio è che si possa identificare gli oggetti e interagire con essi nel codice mentre sono nascosti all'utente. Ciò può condurre a falsi negativi.	Le immagini possono essere ridimensionate in base alle dimensioni dello schermo, ma i test falliranno non appena il layout cambia.
Esperienza utente	Di solito sono richiesti script manuali, almeno per migliorare quelli registrati per leggibilità e manutenibilità.	Test completamente basati sull'interfaccia grafica senza la necessità di script.
Velocità di esecuzione	Tende ad essere più veloce del confronto di immagini/OCR, specialmente quando si utilizzano strumenti nativi forniti dal produttore del sistema.	Tende ad essere più lento a causa della necessità di confrontare lo schermo pixel per pixel con un'immagine di riferimento.

Manutenzione	Dipende dalla qualità degli script di test.	Principalmente consiste nel fornire immagini di riferimento modificate.
Authoring challenge	Conoscenza richiesta del linguaggio di scripting e dei metodi di progettazione del software per costruire una soluzione di automazione sostenibile.	Generazione di immagini di riferimento, soprattutto quando l'app cambia spesso.

5.3 Valutazione degli Strumenti di Automazione

Per avere successo nella creazione di soluzioni di automazione dei test, i team che si occupano di questo aspetto devono scegliere un set di strumenti adeguato. È necessario prendere in considerazione le principali differenze tra gli strumenti disponibili e la loro idoneità ai requisiti del progetto (vedere anche [ISTQB_CTAL_TAE_2016]).

I parametri di valutazione per gli strumenti di automazione dei test possono essere suddivisi in due categorie:

- Adeguatezza organizzativa
- Adeguatezza tecnica

I parametri di adeguatezza organizzativa sono descritti nel capitolo 6.2 del syllabus ISTQB® Foundation Level [ISTQB_CTFL_2018].

I parametri di adeguatezza tecnica includono quanto segue:

- Requisiti dell'automazione dei test e complessità tecniche associate, come l'uso da parte dell'app di nuove funzionalità tipo FaceID, impronte digitali e chatbot.
- Requisiti di ambiente dei test, come condizioni di rete variabili, importazione o creazione di dati di test e virtualizzazione lato server.
- Funzionalità di test reporting e feedback loop.
- La capacità del framework di gestire e guidare l'esecuzione su larga scala sia localmente che in un laboratorio di test nel cloud.
- Integrazione del framework di test con altri strumenti utilizzati nell'organizzazione.
- Disponibilità di supporto e documentazione per aggiornamenti attuali e futuri.

5.4 Approcci per l'Allestimento di un Laboratorio per i Test Automatizzati

Durante l'esecuzione dei test delle applicazioni mobili, gli sviluppatori e i tester posso scegliere tra diverse opzioni per l'allestimento del laboratorio di test dei dispositivi al fine di indirizzare le proprie esigenze di automazione di test.

- Laboratorio di test dei dispositivi on-premise
- Laboratorio di test dei dispositivi remoto

È possibile applicare varie combinazioni di questi approcci. Le loro caratteristiche principali sono descritte e confrontate nel paragrafo 4.4.

I laboratori di test dei dispositivi on-premise sono generalmente difficili da gestire e richiedono molto tempo per la manutenzione. Avere dispositivi localmente in parallelo con emulatori e simulatori servirebbe al meglio le esigenze delle prime fasi di sviluppo e testing dell'app mobile.

Quando si raggiunge una fase più avanzata dello sviluppo dell'app, i team devono eseguire test di regressione completi, test funzionali e test non-funzionali. Questi test vengono eseguiti al meglio su un laboratorio dei dispositivi completo. È in questa situazione che si gestisce in cloud un laboratorio di test di dispositivi remoto, con i continui aggiornamenti necessari. Tali laboratori remoti completano quello in on-premise e garantiscono che combinazioni sufficienti di dispositivi e sistema operativo siano disponibili e aggiornate. Utilizzando i laboratori di test remoti comunemente disponibili, i team hanno accesso a un set più ampio di funzionalità supportate, tra cui test report più ricchi e funzionalità avanzate di automazione dei test.

Infine, quando si eseguono test su larga scala attraverso un framework di automazione dei test o attraverso un processo di integrazione continua (CI), la stabilità dell'intero laboratorio di test è la chiave per l'efficienza e l'affidabilità dei test stessi. Tali laboratori sono in genere progettati per garantire che dispositivi e sistemi operativi siano sempre disponibili e stabili.

I laboratori di test dei dispositivi remoti non sono sempre necessari nelle fasi successive di sviluppo dell'app. Quelli on-premise, se ben progettati e mantenuti, possono essere buoni o migliori di qualsiasi laboratorio di test dei dispositivi remoto.

6. Riferimenti

6.1 Documenti ISTQB®

- [ISTQB_CTFL_2018]:
ISTQB® Certified Tester – Foundation Level Syllabus – Versione 2018
- [ISTQB_FLAT_2014]:
ISTQB® Certified Tester – Foundation Level Extension Syllabus – Agile Tester – Versione 2014
- [ISTQB_FLUT_2018]:
ISTQB® Certified Tester – Foundation Level Specialist Syllabus – Usability Testing – Versione 2018
- [ISTQB_CTFL_PT_2018]:
ISTQB® Certified Tester – Foundation Level Specialist Syllabus – Performance Testing – Versione 2018
- [ISTQB_CTAL_SEC_2016]:
ISTQB® Certified Tester – Advanced Level Specialist Syllabus – Security Testing – Versione 2016
- [ISTQB_CTAL_TAE_2016]: ISTQB® Certified Tester – Advanced Level Specialist Syllabus - Test Automation Engineer - Versione 2016
- [ISTQB_GLOSSARY]:
ISTQB®'s Glossary of Terms used in Software Testing, Versione 3.2

6.2 Libri di riferimento

- [Knott15] 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 Ulteriori libri e articoli

- 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 Link (Web/Internet)

Dichiarazione di non responsabilità: tutti i collegamenti sono funzionanti in data 5 gennaio 2019

- [URL1] http://gs.statcounter.com/
- [URL2] www.owasp.org
- [URL3] https://developer.android.com/studio/test/monkey
- [URL4] https://www.google.de/amp/s/techcrunch.com/2016/05/31/nearly-1-in-4-people-abandon-mobile-apps-after-only-one-use/amp/
- [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-johnson-2012-heuristics-mnemonics
- [URL9] http://www.kohl.ca/articles/ISLICEDUPFUN.pdf

7. Appendice A - Obiettivi di apprendimento/Livello cognitivo di conoscenza

I seguenti obiettivi di apprendimento si applicano a questo syllabus. Ogni argomento del syllabus verrà esaminato in base all'obiettivo formativo corrispondente.

7.1 Livello 1: Ricordare (K1)

Il candidato riconoscerà, ricorderà e terrà in mente un termine o un concetto.

Parole chiave: identificare, ricordare, recuperare, tenere in mente, riconoscere, conoscere

Esempi:

Può riconoscere la definizione di "fallimento" come:

- “Mancata consegna del servizio a un utente finale o qualsiasi altro stakeholder” o
- "Deviazione del componente o del sistema dalla consegna, servizio o risultato previsti"

7.2 Livello 2: Comprendere (K2)

Il candidato può selezionare le ragioni o le spiegazioni per le dichiarazioni relative all'argomento e può riassumere, confrontare, classificare, categorizzare e fornire esempi per il concetto di test.

Parole chiave: riassumere, generalizzare, astrarre, classificare, confrontare, mappare, contrastare, esemplificare, interpretare, tradurre, rappresentare, dedurre, concludere, categorizzare, costruire modelli

Esempi:

Può spiegare il motivo per cui l'analisi e la progettazione dei test dovrebbero avvenire il prima possibile:

- Per trovare difetti quando è più economico rimuoverli
- Per trovare prima i difetti più importanti

Può spiegare le somiglianze e le differenze tra integrazione e test di sistema:

- Somiglianze: gli oggetti del test sia per i test di integrazione che per quelli di sistema includono più di un componente, e sia i test di integrazione che quelli di sistema possono includere tipi di test non funzionali
- Differenze: i test di integrazione si concentrano su interfacce e interazioni e i test di sistema si concentrano su aspetti dell'intero sistema, come l'elaborazione end-to-end

7.3 Livello 3: Applicare (K3)

Il candidato può selezionare la corretta applicazione di un concetto o di una tecnica e applicarla a un determinato contesto.

Parole chiave: implementare, eseguire, utilizzare, seguire una procedura, applicare una procedura

Esempi:

- Può identificare valori limite per partizioni valide e non valide
- Può selezionare casi di test da un determinato diagramma di transizione di stato per coprire tutte le transizioni

Riferimento (per i livelli cognitivi degli obiettivi di apprendimento):

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 - Glossario dei termini specifici del settore

Termine del glossario	Definizione
2G	Tecnologia di telecomunicazione wireless mobile di seconda generazione.
3D Touch	Vedere "Force touch"
3G	Tecnologia di telecomunicazione wireless mobile di terza generazione.
4G	Tecnologia di telecomunicazione wireless mobile di quarta generazione.
5G	Tecnologia di telecomunicazione wireless mobile di quinta generazione.
ADB	Android Debug Bridge (ADB) - strumento da linea di comando che consente la comunicazione con un dispositivo.
app advertisement-based	Un modello di monetizzazione delle app in cui le organizzazioni di sviluppo guadagnano denaro attraverso le pubblicità mostrate all'interno dell'app.
Android Device Monitor (ADM)	Uno strumento autonomo che fornisce un'interfaccia utente per gli strumenti di debug e analisi delle app Android.
Android Studio	L'ambiente di sviluppo integrato ufficiale (IDE) per Android. Android Studio offre strumenti per la creazione di app su ogni tipo di dispositivo Android.
app shortcut (scorciatoia)	Un collegamento a una serie specifica di azioni definite in un'applicazione dagli sviluppatori di applicazioni su Android 7.1 o versioni successive.
application store	Una piattaforma di distribuzione di applicazioni in cui gli sviluppatori possono caricare le proprie app e gli utenti possono cercarle per scaricarle e installarle sulla propria piattaforma.
aspect ratio (fattore di forma)	Il rapporto tra larghezza e altezza di un display o di un'immagine.
asynchronous communication (comunicazione asincrona)	Un tipo di comunicazione in cui i dati possono essere trasmessi in modo intermittente anziché in un flusso costante.
AVD	Acronimo di Android Virtual Device (dispositivo virtuale Android).
backend system (sistema di backend)	Un sistema server che fornisce funzionalità per altri sistemi.
background app (app in background)	Un'app in esecuzione in background.
backward compatibility (compatibilità all'indietro)	La capacità di un'app di funzionare su versioni precedenti di piattaforme.
barcode (codici a barre)	Una rappresentazione dei dati ottica e leggibile da una macchina.
basic phone (telefono base)	Un telefono cellulare con funzionalità minime come effettuare chiamate, memorizzare numeri di telefono, inviare SMS, orologio e sveglia.

Termine del glossario	Definizione
blind spot (punto cieco)	Una posizione senza rete di telecomunicazione wireless.
blocking/do-not-disturb mode (modalità di blocco/non disturbare)	Una modalità operativa di dispositivi mobili che può essere attivata dall'utente per sopprimere determinate funzionalità: notifiche comuni e chiamate vocali.
Bluetooth	Una tecnologia di comunicazione wireless a corto raggio.
byte-code	Un set di istruzioni progettato per l'esecuzione efficiente da parte di un interprete di software. Chiamato anche codice portatile o p-code.
cellular data (rete dati)	Dati trasferiti su una rete cellulare.
cellular network (rete cellulare)	Una rete cellulare è una rete creata da più celle indipendenti ma connesse.
companion device (dispositivo associato)	Un dispositivo progettato per funzionare in collaborazione con un dispositivo intelligente dipendente.
CPU frequency (frequenza della CPU)	La frequenza di clock del processore.
cross-platform development framework (framework di sviluppo multipiattaforma)	Un framework per sviluppare un'app per varie piattaforme utilizzando la stessa base di codice.
CRUD	Mnemonica per Create/Read/Update/Delete (Crea/Leggi/Aggiorna/Elimina) che viene applicata ai dati.
data integrity (integrità dei dati)	Accuratezza e coerenza dei dati durante l'intero ciclo di vita, inclusi archiviazione, elaborazione e recupero.
data synchronization (sincronizzazione dei dati)	Il processo di portare i dati nello stesso stato attraverso due o più fonti.
data validation (validazione dei dati)	La valutazione della correttezza, accuratezza, omogeneità e utilità dei dati.
dead spot (punto morto)	Vedere "blind spot".
device fragmentation (frammentazione dei dispositivi)	La diversità dei dispositivi disponibili, la loro configurazione hardware unica e il loro impatto sulle app e l'esperienza dell'utente.
DPI/PPI	Acronimo di Dots/Pixels Per Inch (punti/pixel per pollice) - un numero che esprime la densità di un display, in punti o pixel.
emulatore (emulator)	Un'applicazione software che imita il comportamento dell'hardware.
enterprise app (app aziendale)	Un'applicazione creata per essere utilizzata all'interno di un'organizzazione e non destinata all'uso pubblico.
external memory (memoria esterna)	Una memoria aggiuntiva che viene aggiunta al dispositivo tramite un'interfaccia standard. Attualmente le schede SD sono più comuni per i dispositivi mobili.
fat client	Nelle applicazioni client/server, un client progettato per gestire parte o la maggior parte dell'elaborazione dei dati.
feature phone	Una classe di telefoni cellulari che offre più funzioni di un telefono base, ad esempio un browser, ma non offre la piena funzionalità di uno smartphone.
flat file	Un file senza gerarchia interna.

Termine del glossario	Definizione
flight mode (modalità aereo)	Una modalità operativa speciale per dispositivi mobili in cui i trasmettitori radio sono disattivati per evitare interferenze con i sistemi di volo/comunicazione.
Force touch	Una tecnologia sviluppata da Apple Inc. che consente ai trackpad e ai touchscreen di distinguere tra le diverse quantità di forza applicate alle loro superfici.
foreground app (app in foreground)	Un'app eseguita in primo piano sul dispositivo per l'interazione diretta dell'utente.
freemium app (app freemium)	Un modello di business in cui gli utenti non pagano nulla per scaricare l'app e vengono offerti acquisti in-app opzionali.
gesture	Un modello specifico di interazione, come un pinch o uno scorrimento (swipe), per attivare funzioni definite del dispositivo. Ad esempio, un pinch viene comunemente utilizzato per ingrandire e rimpicciolire lo schermo del dispositivo intelligente.
Globalization (globalizzazione)	Vedere internationalization.
GPS	Acronimo di Global Positioning System - in tutto il mondo, una rete di satelliti che invia segnali di tempo. Includendo il segnale di almeno 3 satelliti un ricevitore può calcolare la sua posizione relativa ai satelliti tramite triangolazione.
GSM	Acronimo di Global System for Mobile Communications, originariamente Groupe Spécial Mobile, è uno standard sviluppato dall'Istituto europeo per le norme di telecomunicazione (ETSI) per descrivere i protocolli per le reti cellulari digitali di seconda generazione utilizzate dai dispositivi mobili. Attualmente lo standard più comune al mondo per la comunicazione mobile.
GSM cell (cella GSM)	Una parte di una rete GSM che può essere identificata dal suo ID cella univoco
hybrid app (app ibrida)	Un'applicazione che combina tecnologie native e web. Solitamente un'app ibrida utilizza un frame nativo da installare sul dispositivo per interagire con le librerie del dispositivo e simili. Inoltre, viene mostrato il contenuto ricevuto da un server Web.
I18N	Numeronimo (parola basata sui numeri) per internazionalizzazione.
IDE	Integrated Development Environment (Ambiente di sviluppo integrato): un'applicazione software che offre servizi completi ai programmatori di computer per lo sviluppo del software.
in-app purchase (acquisti in-app)	Contenuti e funzionalità extra acquistabili direttamente da un'app.
instruments	Uno strumento di analisi e test delle prestazioni incluso come parte del set di strumenti Xcode.
internal memory (memoria interna)	Una memoria inclusa nell'hardware del dispositivo.
internationalization (internazionalizzazione)	Il processo di preparazione di un'applicazione per adattarsi a varie versioni localizzate.
interrupt	Un evento che si verifica durante un altro evento.

Termine del glossario	Definizione
IoT appliance (apparecchio IoT)	Internet of Things (Internet delle Cose). Un dispositivo o, ad esempio, un sensore di interesse connesso a Internet.
jailbreaking	Una escalation di privilegi allo scopo di rimuovere le restrizioni del software imposte da un sistema operativo. Termine generalmente utilizzato su iOS. Simile al rooting per Android.
L10N	Numeronimo per localizzazione.
landscape mode (modalità paesaggio)	L'orientamento del dispositivo in cui la larghezza del display è maggiore dell'altezza.
library (libreria)	Una raccolta di risorse non volatili utilizzate da programmi per computer, ad esempio funzioni dell'applicazione.
localization (localizzazione)	Il processo di adattamento di un'app o di un prodotto a una determinata regione mediante azioni come la traduzione e gli adattamenti del formato.
look and feel (aspetto e percezione)	Impressione visiva ed emotiva di qualcosa.
Mnemonic (Mnemonica)	Un aiuto per la memoria.
Mobile Application Testing	Testing di app mobili.
mobile device type (tipo di dispositivo mobile)	Una classificazione dei dispositivi mobili in base alle loro caratteristiche di base. Le classi comuni includono basic phone (telefono base), feature phone, smartphone, phablet, tablet e dispositivi indossabili.
mobile emulator (emulatore mobile)	Rappresentazione virtuale di una piattaforma hardware. Ad esempio, l'emulatore Android è un hardware virtuale che esegue una vera immagine del sistema operativo Android. La stessa immagine del sistema operativo potrebbe essere distribuita all'hardware e funzionerà, poiché è il sistema operativo reale.
mobile OS (SO mobile)	Un sistema operativo appositamente progettato per dispositivi mobili.
mobile platform (piattaforma mobile)	Un ecosistema attorno a un sistema operativo mobile, che di solito include strumenti di sviluppo, il sistema operativo stesso e un canale di distribuzione delle applicazioni.
mobile space (spazio mobile)	Un termine che sintetizza tutto ciò che riguarda la tecnologia dei dispositivi mobili, dal mercato e suoi player ai dispositivi e alle app.
mobile simulator (simulatore mobile)	Un ambiente di runtime virtuale. Ad esempio, il simulatore iOS finge di essere iOS ma in realtà non è un vero iOS.
multi-platform applications (applicazioni multipiattaforma)	Applicazioni progettate e sviluppate per funzionare su più piattaforme utilizzando la stessa base di codice per tutte le piattaforme.
multi-tier	Un approccio di progettazione di applicazioni back-end in cui 2 o più server offrono funzionalità specializzate.
multi-touch	Un tipo di interazione con un dispositivo che utilizza vari eventi touch in parallelo.
native app (app nativa)	Un'applicazione sviluppata appositamente per una determinata piattaforma, in genere utilizza API della

Termine del glossario	Definizione
	piattaforma e strumenti di sviluppo forniti dalla piattaforma stessa.
NFC	Near Field Communication - una tecnologia di comunicazione radio a corto raggio.
notification (notifica)	Un avviso inviato dal dispositivo.
OCR	Optical Character Recognition. Il riconoscimento di immagini di testo contenute in un'immagine elettronica e la sua conversione in testo machine-encoded.
on-premise lab (lab on-premise)	Un laboratorio che si trova fisicamente nello stesso posto dell'utente del laboratorio.
orientation (orientamento)	Il posizionamento di un oggetto nel cellulare comunemente usato per esprimere il modo in cui viene utilizzato il dispositivo. Può essere orizzontale o verticale.
OTA	Over the air. Trasmissione dei dati tramite segnali radio, comunemente utilizzati per fare riferimento all'installazione di app su un dispositivo direttamente da una sorgente non collegata via cavo.
overflow	Una situazione in cui i dati in entrata superano ciò che può essere ospitato.
paid app (app a pagamento)	Un'app che viene monetizzata vendendola negli app store.
Persona	Un modello/archetipo per un determinato gruppo di utenti.
portrait mode (modalità ritratto)	Un orientamento del dispositivo in cui l'altezza del display è maggiore della larghezza.
power consumption (consumo energetico)	La quantità di energia consumata.
power save mode (modalità risparmio energetico)	Una modalità operativa di dispositivi mobili che può essere attivata dall'utente o dal dispositivo stesso per risparmiare energia.
power state (stato di consumo di energia)	Un profilo definito dall'utente o predefinito per quanto riguarda il consumo di energia che può essere attivato su un dispositivo mobile.
preferences (preferenze)	I parametri generali di configurazione del dispositivo o dell'applicazione che possono essere modificati dall'utente.
pre-installed app (app preinstallata)	Un'applicazione mobile installata dal produttore del dispositivo. Di solito l'utente non è in grado di disinstallare queste applicazioni.
QR code (Codice QR)	(Abbreviato da Codice a risposta rapida - Quick Response Code) è il marchio commerciale di un tipo di codice a barre a matrice (o codice a barre bidimensionale).
remote device access (accesso ai dispositivi remoti)	Interazione con un dispositivo fisicamente situato in una posizione diversa rispetto al suo utente, di solito su Internet.
retain app data (conserva i dati dell'app)	I dati generati dall'utente o il contenuto dell'app vengono conservati sul dispositivo quando l'applicazione viene disinstallata per essere accessibile da altre app o quando viene reinstallata la stessa app.

Termine del glossario	Definizione
rooting	Il processo per ottenere l'accesso come root al sistema operativo del dispositivo. Il termine viene solitamente utilizzato sulla piattaforma Android. Simile al jailbreak su iOS.
runtime environment (ambiente di runtime)	Implementazione del modello di esecuzione. Conosciuto anche come sistema di runtime.
screen real-estate	La quantità di spazio fornita dal display.
software development kit (SDK)	Un set di strumenti e librerie per sviluppare software per una determinata piattaforma.
sensitive data (dati sensibili)	Dati che richiedono una protezione speciale, come password e dati personali.
sensor (sensore)	Un dispositivo, un modulo o un sottosistema il cui scopo è rilevare eventi o cambiamenti nel suo ambiente e inviare le informazioni ad altri dispositivi elettronici, spesso un computer processor.
session (sessione)	Un evento time-boxed (periodo di tempo di determinata durata).
session sheet (foglio di sessione)	Un documento per focalizzare l'ambito di una sessione di test e registrarla.
side-loading	Il caricamento/l'installazione di un'applicazione tramite un mezzo diverso da un application store.
single-tier	Un approccio di progettazione di applicazioni back-end in cui un singolo server fornisce tutti i servizi necessari per un'applicazione.
smartphone	Un personal computer palmare con un sistema operativo mobile e una connessione di rete cellulare mobile a banda larga integrata per voce, SMS (servizio di messaggistica breve; spesso indicato come testo) e comunicazione di dati su Internet.
soft keyboard (tastiera software)	Una tastiera virtuale realizzata in software, presentata all'utente su un display.
store-and-forward	Un approccio di sincronizzazione dei dati in cui i dati vengono archiviati localmente e inoltrati al server quando esiste una connessione di rete appropriata.
synchronous communication (comunicazione sincrona)	Un metodo di trasferimento dei dati in cui i flussi di dati vengono inviati (a monte) e ricevuti (a valle) alla stessa velocità e sono distanziati dai segnali di temporizzazione.
tablet	Un tipo di dispositivo mobile, comunemente utilizzato per dispositivi con schermi da 7" e oltre.
thin client	Nelle applicazioni client/server, un client progettato per essere particolarmente piccolo in modo che la maggior parte dell'elaborazione dei dati avvenga sul server.
third-party marketplace (marketplace di terze parti)	Una piattaforma di distribuzione di app non gestita da un fornitore di piattaforme.
transaction-based apps (app transaction-based)	Un'applicazione in cui l'utente paga per transazione.
upload conflict (conflitto di caricamento)	Errore durante il tentativo di caricare un file che è già presente nella destinazione di caricamento.

Termine del glossario	Definizione
wiewport size (dimensioni del viewport)	Una dimensione dello schermo virtuale utilizzata dal browser per adattare il layout allo schermo.
VPN (Virtual Private Network)	Un canale privato encrypted (crittografato) tramite una rete pubblica.
wearable (dispositivo indossabile)	Un dispositivo informatico indossato sul corpo come un orologio o degli occhiali.
web app (app web)	Un'applicazione ospitata su Internet utilizzata tramite un browser.
Xcode	Un ambiente di sviluppo integrato fornito da Apple per sviluppare applicazioni OSX e iOS.

9. Indice