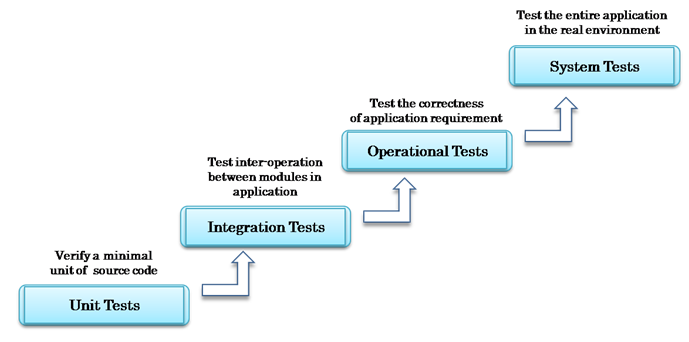
**Testing for Android**

Il testing è un processo utilizzato per individuare l’affidabilità, la completezza e la complessità dei componenti software in corso di sviluppo. Una corretta strategia di test per Android dovrebbe includere i seguenti passi:

* Unit test
* Test d’integrazione
* Operational Test
* Test del sistema



## Unit test

Per unit test si intende l’attività di testing effettuata su singole unità del software dotate di comportamento autonomo, che possono essere delle specifiche funzioni o singole classi. Per l’unit test è necessaria la scrittura di driver e stub. Il driver simula un'unità chiamante mentre lo stub simula un'unità chiamata. L'investimento di tempo necessario per questa attività talvolta fa sì che venga ridotto il livello di priorità del test di unità e, quasi sempre, ciò si rivela un errore. Infatti, sebbene lo sviluppo di driver e stub costi in termini di tempo e denaro, il test di unità offre alcuni vantaggi, come l'automazione del processo di test, riduce le difficoltà di rilevamento degli errori contenuti in parti più complesse dell'applicazione e spesso viene ottenuta una maggiore copertura di tutti gli aspetti da verificare, poiché l'attenzione viene posta sulle singole unità. I driver e gli stub possono essere riutilizzati per eseguire frequentemente test sulle continue modifiche che si verificano nel ciclo di sviluppo, senza tuttavia generare altro codice per il test. In questo modo si riduce il rapporto costi/benefici della scrittura di driver e stub e si è in grado di gestire meglio il costo di test ripetuti.

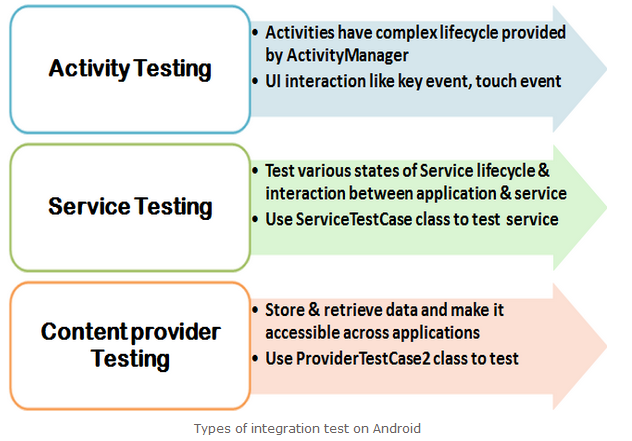
La piattaforma Android integra il framework JUnit 3.0, ma vi sono altri vari strumenti che possono essere utilizzati:

* Robolectric: permette di effettuare test con uno stile più vicino al test black box, direttamente dalla JVM, senza la necessità di un dispositivo o di un emulatore, rendendo il test più efficace per il refactoring e permettendo di eseguire i test Android nell’ambiente di continuous integration senza alcuna configurazione aggiuntiva. In sostanza, Robolectric permette di avere iterazioni rapidamente, di scrivere test in stile black box, di avere test sul comportamento invece che sull’esecuzione e avere un’alta copertura di test. Per Robolectric, le classi Android sono sostituite da oggetti shadow: se un metodo è implementato da Robolectric, le chiamate al metodo vengono inoltrate all'oggetto shadow che si comportano in modo simile agli oggetti delle SDK di Android. Se un metodo non è implementato dall'oggetto shadow, restituisce semplicemente un valore predefinito. Le release di Robolectric sono disponibili su Maven Central.
* Hamcrest: è un framework che aiuta la scrittura di test del software nel linguaggio di programmazione Java, definendo delle regole 'match' in modo dichiarativo.
* JMock: è una libreria che supporta lo sviluppo test-driven di Java codice con oggetti mock. La libraria jMock rende veloce e facile da definire gli oggetti mock, consente di specificare con precisione le interazioni tra gli oggetti, è efficace per il refactoring ed è facile da estendere.
* Mockito: è un popolare framework che può essere utilizzato sia in combinazione con JUnit che direttamente nei test di Android; in generale, consente di creare e configurare oggetti mock, anche se tiene alcune limitazioni poiché non può testare le classi finali, le classi anonime e i tipi primitivi.
* RoboGuice: è un dependency injection framework per applicazioni Android che dovrà integrarsi con le classi native della piattaforma. RoboGuice ha l’obiettivo di semplificare l'installazione di applicazioni Android, come view, system services, resource.
* Spring: framework che permette di utilizzare RestTemplate, un popolare client REST basato su Java, per le applicazioni Android. Spring fornisce anche il supporto per l'integrazione di funzionalità di Spring Social nell’applicazione Android, che include OAuth, client per l’autorizzazione e l’implementazione di popolari siti web di social, come Twitter e Facebook.
* Robotium: framework open-source di test, rende più facile scrivere test automatici black-box efficaci per le applicazioni Android. Con il supporto di Robotium, gli sviluppatori del test case possono scrivere scenari di test di funzionalità, di sistema e di accettazione, che coprono molteplici activity Android. Robotium permette di scrivere in breve tempo i casi di test e si integra senza problemi con Maven, Gradle o Ant per eseguire i test come parte di integrazione continua.

## Integration Test

Il test di integrazione rappresenta l'estensione logica del test di unità. Il tipo più semplice di test di integrazione consiste nella combinazione di due unità già sottoposte a test in un solo componente e nel test dell'interfaccia presente tra le due. In questo caso, per componente si intende l'aggregazione integrata di più unità. In uno scenario realistico più unità vengono combinate in componenti che, a loro volta, vengono aggregati in parti più grandi del programma. Il concetto che è alla base di questo approccio consiste nell'esecuzione del test delle combinazioni di parti ed nell'espansione del processo al test dei moduli di un gruppo con quelli di altri gruppi. Alla fine, tutti i moduli che compongono un processo vengono sottoposti al test contemporaneamente. Questo metodo consente di ridurre notevolmente il numero di possibilità e di semplificare in larga misura l'analisi.

In Android, il test di integrazione spesso comporta il controllo di componenti Android come i testing sui servizi, sulle activity o sui Content Provider. La documentazione ufficiale di Android tende a concentrarsi su test basati sul AndroidTestCase e le sue sottoclassi, ma in realtà vi sono molti framework che permettono di condurre test di integrazione per Android come Roboelectric, Robotium (visti precedentemente) e Troyd, framework utile per generare test basati su scenari attraverso comandi tester e un'interfaccia a riga di comando.



## Test operativo

Il testing operativo permette di valutare un'applicazione software prima della fase di produzione, verificando il prodotto dal punto di vista delle funzionalità. In Android, FitNesse è un framework open-source che consente ai clienti, tester e programmatori di imparare ciò che il programma dovrebbe fare , e di confrontare automaticamente con ciò che effettivamente deve fare. Esso mette a confronto le aspettative dei clienti ai risultati effettivi.

## Test di sistema

In questa fase di testing, il sistema viene testato nel suo complesso e viene verificata l'interazione tra i componenti, software e hardware. In Android, test di sistema include normalmente test GUI, test di usabilità e test sulle prestazioni. È possibile utilizzare strumenti come Traceview per condurre test delle prestazioni su Android. Vi sono altri strumenti per il test di sistema, oltre a Robotium, come Monkeyrunner, realizzato essenzialmente per testare applicazioni e dispositivi a livello funzionale e per l'esecuzione di unit test suite.