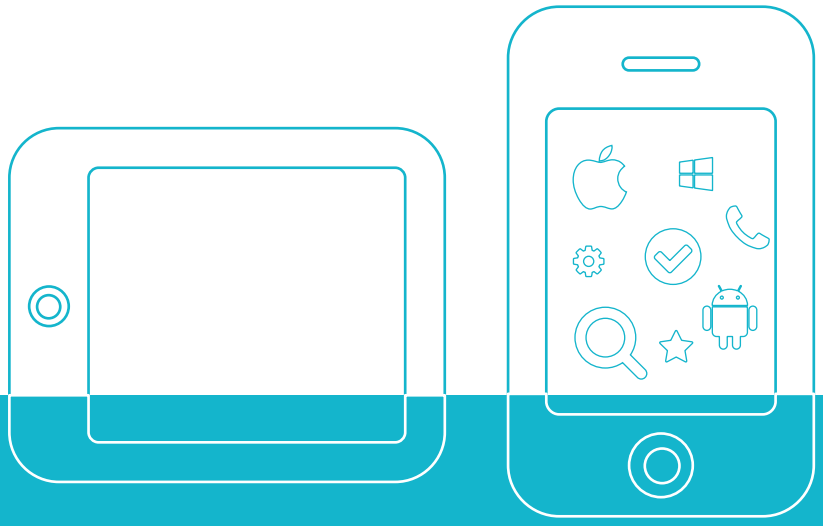
kanda SOFTWARE



AUTOMATING MOBILE APPLICATION TESTING

Reducing testing costs while raising the quality of your app

CONTACT US

Kanda Software
200 Wells Avenue,
Newton MA 02459

contact@kandasoft.com

617-640-3850

Mobile application users are savvy and have high expectations for the quality for the applications they install on their devices. Users expect mobile apps to be responsive, stable and secure. They want simple yet comprehensive user interfaces with a bug-free functionality. You need to meet those demands across multiple operating systems including iOS, Android, Windows, and multiple devices.

In today's world, it's a challenge for any company to achieve and maintain the quality of mobile applications. Many companies neglect mobile testing, because it imposes additional time and cost constraints. The lack of mobile testing strategy results in a lower application quality and high abandonment rates and, eventually, in overall increased quality-associated costs

Mobile application test automation can significantly improve development process and, will quickly pay off. Automated tests can run frequently and are well suited when testing in an agile environment that requires the ability to quickly adjust to changing mobile development goals and processes. New test cases are generated continuously and can be added to existing automation processes without interfering with the development process.
It is important to understand that 100% mobile test automation is hardly realistic. When making a decision which tests should be automated, their value and the required effort need to be compared. High-value/Low-effort tests should be automated first.

In general, the most appropriate tests for automation are:

- Repetitive tests that run for multiple builds
- Tests that tend to cause human error
- Tests that require multiple data sets
- Frequently used functionality and compatibility tests
- Tests that require significant effort and time when manual testing.
- Normalized testing – especially those for performance, responsiveness and throughput

Challenges in Mobile App Test Automation

#1 Mobile Platforms and Technology Coverage

The primary factor that determines mobile test automation strategy success, is the ability of the chosen tool to work across various platforms and technology stacks. The following challenges influence automation success:

Device Diversity:

- Multiple platforms and browsers.
- Rendering differences.
- Mobile devices with different application runtimes.

Network:

- Multiple network types (GSM/GPRS/Wi-Fi, etc.).
- Different speeds of connectivity across geographic regions
- Multiple network operators with customized network features

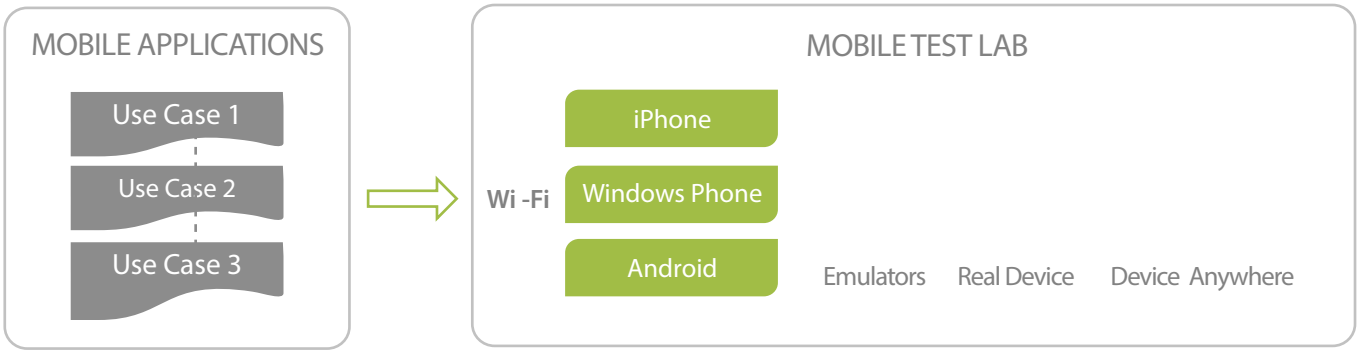
Hardware:

- Limitations in processing speed
- Limitations of mobile memory size
- Differences in device communication protocols

#2 Test Execution Challenge

It is a challenging task to execute tests across different devices, OS' and browsers. Test automation procedures require designing a test execution matrix that can be both time-consuming and difficult. For example, consider a test case for the mobile application running on iOS, Android and Windows devices. In this case, the test has to be executed on 13* devices and in 10* different combinations, which leads to 130* rounds of testing. (see Figure 1)

Real-time Scenario for a Smartphone-based App: Sample Test Execution



TIPS TO GET STARTED WITH AUTOMATED MOBILE TESTING

Successful mobile test automation required careful planning.

First, create a Mobile App Automation plan: identify the initial set of tests appropriate for automation; choose what types of tests you need to ensure high quality of the mobile app.

Second, decide what actions your automated tests will perform. Complex automated tests are difficult to debug and edit, so divide your tests into smaller logical parts. When enhancing mobile app functionality, it will be easier to update automated testing environment with the new tests that address the changes.

For the mobile app automation to be successful, follow the following simple rules:

1. Decide what Test Cases to Automate

2. Test Early and Test Often

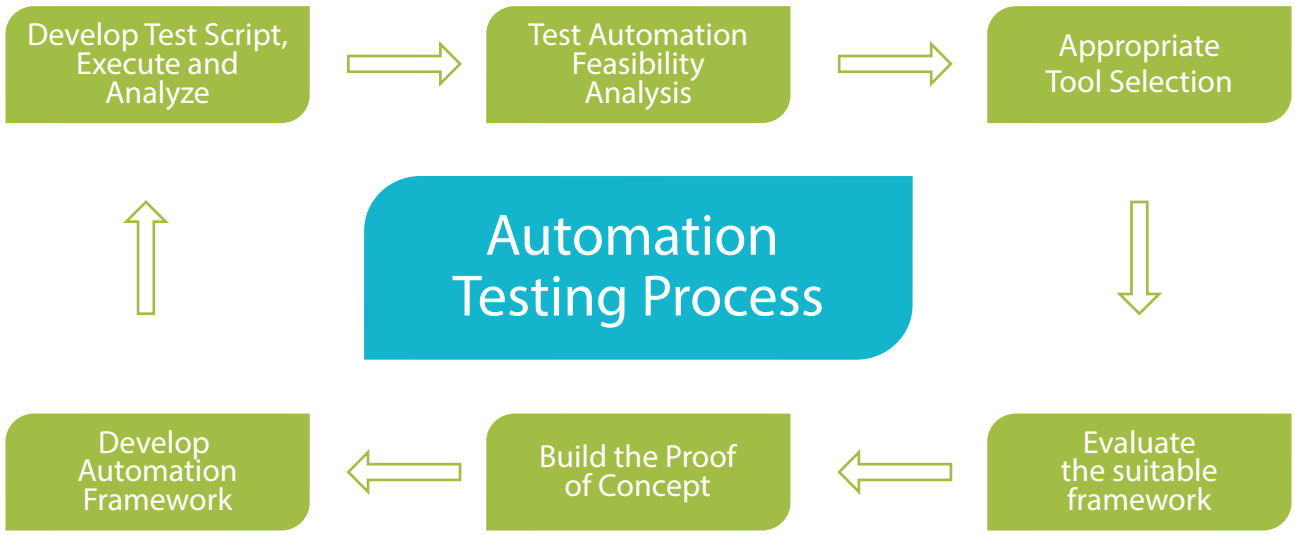
3. Select the Right Automated Testing Tool

4. Divide your Automated Testing Efforts

5. Create Good, Quality Test Data

6. Create Automated Tests that are Resistant to Changes in the UI

Mobile Application Test Automation Process:



Mobile Testing Automation Tools

There're a lot of mobile test automation tools on the market. It is essential to choose the tool that best meets company's overall requirements and takes the nature of the mobile application (native, HTML 5, hybrid) into account.

When selecting a test automation tool consider the following:

- The tool must support mobile application OS, language and technology stack
- It should be modular and easily upgradable to prevent lack of support from the test automation platform
- Consider feature set of the tool and the usability. Is there a capability of manually creating automated tests, record-and-playback test creation support, etc.?
- A blend of technical and non-technical team members should be able to create, maintain and execute scripts.
- The tool should support creation and maintenance of individual test cases, but also the creation, maintenance and management of whole test sets.

- The tool should have a capability of the dependences and modularity management, e.g. being able to define interfaces to user interface elements only once, so that changes to the application can be reflected by changing only this interface in the test automation set.
- It should support the management of test data separately from the automation scripts so that test data sets can be changed with little overhead.
- Test case execution on multiple mobile device platforms is optimal, so that a single automation environment can be used to test all relevant devices.

Mobile Test Automation Tool Categories:

	Native Platform Frameworks	Multi-Platform Tools	
		Visual Based	Object Based
Description	• Provided by mobile platform vendor as part of SDK • Typically interact with the application at the UI object level	• Interacts with devices using visual methods (text and image recognition)	• Interacts directly with application UI objects using native methods and properties
Pros	• Object based interaction with app UI controls • Greatest degree of support for native UI objects • Control of device settings	• Supports multiple platforms • Ability to execute tests across multiple devices • Interact with entire device	• Supports multiple platforms • Ability to execute tests across multiple platforms • Interacts with apps at the object level for greater reliability
Cons	• Limited to a single platform • Requires code level access to the app • May be limited to simulator (e.g. Blackberry)	• Generally less reliable • No access to object properties • May require jail-breaking / rooting	• Interaction is limited to app under test • Requires app to be instrumented
Example	• Android UI Automator • iOS UI Automation	• SeeTest (Experitest) • eggPlant (TestPlant) • Perfecto Mobile • Zap-Fix (Zap Tech.)	• Trust (Mobile Labs) • TouchTest (SOASTA) • M-eux Test (Jamo Solutions)

Popular Mobile Test Automation tools

Tool	Paid/Open Source	Native Apps	Web	Hybrid Apps	Android	IOS	Windows	Blackberry	Library/ Tool
Robotium	Open Source	Y	-	Y	Y	-	-	-	Library
Sikuli	Open Source	Image Based	Image Based	Image Based	Y	Y	Y	Y	Both
Selenium WebDriver	Open Source	-	Y	-	Y	Y(but obsolete)	-	-	Library
NativeDriver	Open Source	Y	-	-	Y	Y	-	-	Library
Appium	Open Source	Y	-	Y	Y	Y	-	-	Tool
MonkeyTalk	Open Source	Y	Y	Y	Y	Y	-	-	Tool
SeeTest	Paid	Y	Y	Y	Y	Y	Y	Y	Tool
M-eux (JamoSolutions)	Paid	Y	-	Y	Y	Y	Y	Y	Tool
EggPlant	Paid	Image Based	Image Based	Image Based	Y	Y	Y	Y	Tool
mAutomate	Paid	Y	Y	Y	Y	Y	-	-	Web Based
Ranorex	Paid	Y	Y	Y	Y	Y	-	-	Tool

Free and open source automation tools are an attractive option for smaller companies with relatively simple mobile app functionality and interfaces. Usually, these tools come with the reduced functionality. For example, Robotium for Android is not comprehensive enough to cover all relevant requirements for mobile test automation.

It only covers a small subset of user interface elements (such as buttons, lists, input elements, etc). Applications that use sophisticated UI elements will need custom-built frameworks on top of Robotium to enhance the tool to cover the required functionality.

Conclusion

Mobile app Testing automation is an effective way of expediting the quality assurance process and reducing costs. Although, some of the tests, including GUI testing can only be performed manually, the majority of compatibility and functionality tests can effectively be automated. Factors, such as support for applicable mobile platforms, script reusability and total cost of ownership should be taken into account when selecting the right mobile test automation tool.

ABOUT KANDA

Kanda has been developing mobile applications and responsive UIs for a variety of clients for nearly 5 years. We are constantly pushing the boundaries of what's possible on mobile. We make sure that your idea is transformed into a beautifully designed, user friendly and engaging mobile solution of a superior quality.

In 2014 Kanda was ranked by Inc. Magazine in the Top 500 List of the annual Inc. 500|5000 - an exclusive ranking of the nation's fastest-growing private companies in America. Kanda was named 13th among Massachusetts-based companies and 459th out of 5000 fastest growing US companies for its sustained and rapid three-year growth.

Kanda provides a full spectrum of mobile app development, QA and testing solutions. Some mobile testing services include:

- Functional testing
- Risk-Based testing
- Acceptance, Usability and Accessibility testing
- Performance (Stress and Load) testing

- Regression
- Installation and Configuration
- Security
- API testing
- Smoke tests
- Memory leakage, interrupt, carrier and protocol validation testing
- Functionality testing
- Geo-Fence testing
- And more…

CONTACT US

Kanda Software

200 Wells Avenue, Newton MA 02459

contact@kandasoft.com

617-640-3850