Mobile Testing Trends
What’s in Your Mind
Agenda

1. Mobile Trends
2. Mobile Platforms
3. Mobile Development Methodologies
4. Mobile Application Testing
5. Mobile Testing Approaches
6. Best Practices
**Mobile Device**

A mobile device is a pocket-sized computing device, typically with a display screen with touch input or a miniature keyboard.

Eg) PDA, MID, Tablets, EDA fall under Mobile devices category

**Mobile Networks**

- **First Generation**
  - Analog Transmission
  - Mainly Speech
  - Voice Band Data
  - Circuit Switched
  - Local Systems

- **Second Generation**
  - Digital Transmission
  - Digital Data
  - Global Roaming

- **Third Generation**
  - Digital Transmission
  - Speech and Video
  - Mainly Packet Switched

- **Fourth Generation**
  - Hi speed Digital Data
  - Only Packet Switched
  - VoIP
Mobile Platforms

iOS by Apple
Device: iPhone, iPad, Apple TV

Windows Mobile by MS
Devices: Nokia, HTC and others

Proprietary OS by RIM
Device: Blackberry

Brew by Qualcomm
Devices: Samsung and others. Mainly for CDMA mobiles

Android by Google
Device: Samsung, LG, HTC, Moto & others

Bada from Samsung
Devices: Samsung

Symbian by Nokia
Devices: Nokia, Sony Ericsson

In addition to the above there are less popular proprietary OS as well
# Mobile Platforms – An Overview

<table>
<thead>
<tr>
<th>Platform</th>
<th>Current Version</th>
<th>Source Model</th>
<th>Multi-tasking</th>
<th>Web Browser</th>
<th>Patch Updates</th>
</tr>
</thead>
<tbody>
<tr>
<td>iOS</td>
<td>6.0</td>
<td>Apple</td>
<td>Closed Source</td>
<td>Safari web browser</td>
<td>Sync Patch / OTA</td>
</tr>
<tr>
<td>Android</td>
<td>4.1</td>
<td>Open Handset Alliance</td>
<td>Open Source</td>
<td>Chrome/ Webkit Browser</td>
<td>Synch Patch updates</td>
</tr>
<tr>
<td>Windows Phone</td>
<td>7.5</td>
<td>Microsoft</td>
<td>Closed Source</td>
<td>Internet Explorer</td>
<td>Sync Patch / OTA</td>
</tr>
<tr>
<td>Symbian</td>
<td>Symbian^4</td>
<td>Closed Source</td>
<td>Closed Source</td>
<td>Webkit Browser</td>
<td>Sync Patch / OTA</td>
</tr>
<tr>
<td>BlackBerry OS</td>
<td>7.0</td>
<td>RIM</td>
<td>Closed Source</td>
<td>Webkit Browser</td>
<td>Sync Patch / OTA</td>
</tr>
</tbody>
</table>
Mobile Evolution

1940
Mobile Technology first appeared in Taxi, long-haul trucks and police cars

1983
First Generation 1G was introduced by Motorola and 148,855 worldwide

1985
340,000 Users in USA 750,629 worldwide

1990
5.2 million users in USA 11.1 million users worldwide

2000
110 million users in USA 720 million users worldwide

2011
323 million users in USA 5.9 billion users worldwide

Mobile Users > Desktop Internet Users Within 5 Years

Computing Growth Drivers Over Time, 1960 – 2020E

Increasing Integration

Smartphone Deployment Approach

By Industry Verticals

Smartphone preference in enterprises

13,000+ iPhone apps downloaded

180+ transactions on eBay mobile app

$10,000+ transaction on PayPal mobile

18 Amazon Kindle Fire sold

81 iPad, 925 iPhone sold

103 Blackberry sold
Key Stats:

As of June 2012, Apple leads with more 650,000 Apps with more than 30 billion downloads

More than 600,000 apps available for Android, and the estimated number of applications downloaded from Google Play was 20 billion

By 2015 mobile app development projects will outnumber native PC projects by a ratio of 4-to-1*

Mobile Internet usage is expected to overtake desktop internet usage by 2014 *

By 2015, International Data Corporation (IDC) predicts that 182.7 billion mobile apps will be downloaded. That’s a 1600% increase from the 10.7 billion apps downloaded in 2010.

* http://www.gartner.com
## Worldwide Smartphone operating system (OS) market share in 2009-2015, according to Gartner

<table>
<thead>
<tr>
<th>Operating Systems</th>
<th>2009 market share</th>
<th>2010 market share</th>
<th>2011 market share</th>
<th>2015 market Share</th>
</tr>
</thead>
<tbody>
<tr>
<td>Android</td>
<td>3.9%</td>
<td>22.7%</td>
<td>38.5%</td>
<td>48.8%</td>
</tr>
<tr>
<td>BlackBerry</td>
<td>19.9%</td>
<td>16.0%</td>
<td>13.4%</td>
<td>11.1%</td>
</tr>
<tr>
<td>iOS</td>
<td>14.4%</td>
<td>15.7%</td>
<td>19.4%</td>
<td>17.2%</td>
</tr>
<tr>
<td>Symbian</td>
<td>46.9%</td>
<td>37.6%</td>
<td>19.2%</td>
<td>0.1%</td>
</tr>
<tr>
<td>Windows Phone/Mobile</td>
<td>8.7%</td>
<td>4.2%</td>
<td>5.6%</td>
<td>19.5%</td>
</tr>
<tr>
<td>Others</td>
<td>6.1%</td>
<td>3.8%</td>
<td>3.9%</td>
<td>3.3%</td>
</tr>
</tbody>
</table>

| Total smartphones sold  | 172 million       | 297 million       | 468 million       | 631 million       |
What is Mobile App

Mobile Applications are a rapidly developing segment of the global Mobile Market. They consist of software that runs on a mobile device and performs certain tasks before the user of the Mobile Phone. They can be downloaded physically through USB / WIFI from a desktop or can be downloaded by a web server over internet.
Types of Mobile Applications

Native Applications
Native applications are developed for a specific platform and installed on the device

Hybrid Applications
Hybrid application (hybrid app) is one that combines elements of both native and Web applications

Web Based Applications
Web based applications are accessible through device browser or third party browsers installed on the device

Deciding the type

Depending on the requirements. Some apps are a better fit with web technologies than others. Knowing the pros and cons of each approach will help you make the right decision about which path is appropriate for your situation.
App Development Methodologies
Native Application development

Native App - Native apps are built for a specific platform with the platform SDK, tools and languages, typically provided by the platform vendor

- Best Suited to provide best user experience and for targeted devices
- Dependent on native platform and hence requires separate code bases for respective device platforms

Advantages
- Ability to leverage device-specific hardware and software
- A richer, more compelling user experience
- Ability to run offline

Disadvantages
- Separate version required for different platforms, which requires more cost and time
- Updation of apps tedious compared to web apps

<table>
<thead>
<tr>
<th>Platform</th>
<th>Programming Language</th>
<th>IDE</th>
<th>App Format</th>
<th>Application Stores</th>
</tr>
</thead>
<tbody>
<tr>
<td>Android</td>
<td>Java</td>
<td>Eclipse</td>
<td>.apk</td>
<td>Google Play Store</td>
</tr>
<tr>
<td>iOS</td>
<td>Objective C</td>
<td>XCode</td>
<td>.app</td>
<td>Apple App Store</td>
</tr>
<tr>
<td>Windows Phone</td>
<td>C#, VB.NET</td>
<td>Visual Studio</td>
<td>.xap</td>
<td>Windows Phone Market</td>
</tr>
<tr>
<td>Blackberry</td>
<td>Java</td>
<td>Eclipse</td>
<td>.cod</td>
<td>Blackberry App World</td>
</tr>
</tbody>
</table>
Mobile Web Application

Web App – Mobile Web apps are server-side apps, built with any server-side technology (PHP, Node.js, ASP.NET) that render HTML that has been styled so that it renders well on a device form factor. They are accessible over device native / third party browsers

- Applicable if existing web page functionality to be accessed from wide range of devices
- Maximum interoperability as there is little or no dependency on native device OS

Advantages
- Compatibility – Mobile Websites are generally compatible across browsers
- Upgradability – Mobile Websites can be Updated Instantly and pushed to users
- Time and Cost - Mobile Websites are Easier to and Less Expensive to develop

Disadvantages
- Limited user experience as there will be a latency for every request to the server
- Accessing device features like Camera, Address book, Bluetooth etc is very limited
- Caching of data – only to certain extent
Hybrid apps like native apps, run on the device, and are written with web technologies (HTML5, CSS and JavaScript).

Hybrid App development:

- Development methodology where mobile applications are developed in one common language and which can be deployed to all the popular platforms at once.

- Hybrid apps run inside a native container, and leverage the device’s browser engine (but not the browser) to render the HTML and process the JavaScript locally.

- HTML5 provides mobile device users richer mobile applications and improved usability.

- Different frameworks PhoneGap, JQuery, Sencha Touch, WorkLight, Appcelerator, PhoneGap, and Worklight.

Advantages:
- Write once deploy anywhere – Cross Compatibility
- Saves time and cost
- Offline data storage

Disadvantages:
- Restricted to the native browser capabilities
- No complete access to all the hardware features like Camera, Accelerometer, Bluetooth
## Comparison between different App Types

<table>
<thead>
<tr>
<th></th>
<th>Native</th>
<th>Hybrid</th>
<th>Mobile Web</th>
</tr>
</thead>
</table>
| **Skills/tools needed for cross-platform apps** | • Objective-C  
• Java  
• C  
• C++  
• C#  
• VB.net | • HTML 5  
• CSS3  
• Javascript  
• Mobile development framework (like PhoneGap) | • HTML  
• CSS  
• Javascript |
| **Distribution**       | App Store/Market                              | App Store/Market                             | Internet                                    |
| **Development Speed**  | Slow                                         | Moderate                                    | Fast                                        |
| **Ongoing application maintenance** | Difficult                                    | Moderate                                    | Low                                         |
| **Device access**      | Full access: (Camera, microphone, GPS, gyroscope, accelerometer, file upload, etc...) | Mostly: (Camera, microphone, GPS, gyroscope, accelerometer, file upload, etc...) | Partial access: (GPS, gyroscope, accelerometer, file upload.) |
| **Offline access**     | Yes                                          | Yes                                        | To an extent                                |
| **Advantages**         | Lets you create apps with rich user interfaces and/or heavy graphics. | Combines the development speed of mobile web apps with the device access and app store distribution of native apps. | Offers fast development, simple maintenance, and full application portability. One mobile web app works on any platform. |
| **Disadvantages**      | • Development Time  
• Development Cost  
• Ongoing Maintenance  
• No portability (apps cannot be used on other platforms) | • Can't handle heavy graphics.  
• Requires familiarity with a mobile framework | • Can't handle heavy graphics.  
• Can't access camera or microphone. |
| **Best used for**      | • Games  
• Consumer-focused apps that require a highly graphic interface | • Consumer-focused apps with a moderately graphical interface.  
• Business-focused apps that need full device access. | • General non-game apps.  
• Business-focused apps |
Mobile Application Testing
Mobile application testing is a process by which application software developed for hand held mobile devices is tested for its functionality, usability and consistency.
Mobile Testing – How is it different?

DETAILED DIVERSITY
• Multiple Platforms
• Multiple Browsers
• Rendering differences
• Mobile devices have different application runtimes.

NETWORK CHALLENGES
• Multiple type of Networks (GSM / GPRS / Wi-Fi / Wi-Max etc)
• Unpredictable time taken for data transfer
• Different speed of connectivity across geographies
• Multiple Network Operators with customized Network features

HARDWARE CHALLENGES
• Limitations in processing speed
• Limitations of Memory size of mobile
• Differences in Communication Protocols of devices WAP/ HTTP.
Customers Challenges in Mobile Testing

- Variety of mobile devices in market and multiple manufacturers
- Coping with the short lifecycle of the mobile application in market
- Huge variety of hardware capabilities
- Shorter duration of device life in the market
- Variety of network modes like 2G/3G/4G/Wi-Fi/Wi-Max
- Huge investment and high time frame to set up a test lab
- Most of the tools are image-based comparison tools
- One test tool may not support all platforms versions
- Tools may need jailbreak/rooting that invokes security threat
- Testing on latest technologies like HTML5, NFC, etc.
- High rise of localized app capability
- Choice between simulators and real devices for testing
Mobile Testing - Types

- Analyzing the logs and measure/monitor the performance of mobile devices in a real world environment across different network infrastructures.
- Testing of a product in the actual context in which it will be used.
- Field tests performed validate handset network success in a real world live network environment.
- Test to ensure the stability of the application so that it can be taken for further levels of testing.
- Documented features are implemented in the application and work as expected.
- Test the text, logos and all UI level validation.
- Validate Usability aspects such as Trackball, Touch etc.
- Validate Application behaves as expected and stable during the course of Interruptions.
- Incoming Calls, SMS Notifications, Low Battery, Network related testing such as Low network, Zero Network etc.
- Tailor devices to the unique language.
- Translate user interfaces and provide documentation and support in local language.
- Perform all types of testing in the respective language.
- Testing to ensure that the response time while navigating between different screens and user load handling are according to defined NFR's.
- Validate Security in terms of data transmission, on-device storage, Session handling, Client side injections etc.
- Test to ensure that the basic functionality of the application is not affected due to any changes made.
- Test to ensure that the new feature works fine as expected.

Mobile Testing

Sanity Testing

Field Testing

Localization Testing

Stress/Performance

Regression Testing

Functional/UI Testing

Interruption Testing
Components of Mobile Testing

- Emulators / Simulators
- Mobile Devices
- Testing tools and Accelerators
- Network connectivity
Device emulator / Simulator is a desktop application that emulates mobile device hardware and operating systems, allowing us to test and debug our applications and see how they are working.

- No cost involved – Various OS versions, screen sizes, Device models are available for free download
- Provided by the OEM, Simulators represents 90% of real life however, certain types of testing are not possible (like interruption, network connectivity)
- Easily installable on a normal system – Android, Black Berry, Symbian, Windows Mobile require Windows desktop while, iOS require MAC
- Ideal for functional testing needs
- Some level of development knowledge required to benefit on the features provided

Emulator / Simulator – The Difference

Emulators mimic the software *and* hardware environments found on actual devices
Simulators, on the other hand, only mimic the software environment
Introduction:

- iPhone and iPad are popular devices from Apple. It has its own proprietary operating system, iOS. It's well known for its design and performance.
- Many applications of native, web and hybrid type are available which can be downloaded from Apple iStore.
- Applications for iPhone can be installed both over air and using iTunes application.

Prerequisites:

MAC Environment

Getting Started:

- Deploy Native app into iPhone Simulator using Xcode.
- Xcode - an integrated development environment that manages your application projects and lets you edit, compile, run, and debug code.
- Once the native app is deployed, the app icon appears on simulator, thus aids to access the app directly from simulator.
- Testing in UIAutomation can be done by both developer and tester, it requires knowledge of iPhone development.
- UIAutomation is the automation technique for iPhone. It uses JavaScript as test script to perform automation. This can be done both in device and simulator.
- iPhone Simulator does not emulate an iPhone processor, disk drive, memory constraints.
OVERVIEW

• Android emulator comes as part of the android SDK commonly known as AVD – Android Virtual Device.
• It lets the user to prototype, develop, and test Android applications without using a physical device.

Android Emulators

• The AVD’s are OS version specific and provides the user the flexibility to customize OS version, resolution, skin, sd card size and various other hardware properties to be emulated.
• There are many command line utilities and tools which comes as part of the sdk which makes it easy to debug and interact with emulator

Prerequisites for Android Emulator
• JRE – Java Runtime Environment
• Android SDK

Installing an application

• If the application is available in Google Playstore it can be directly downloaded and installed on to the device.
• If the application is available in ‘.apk’ format, it can be installed using the command, ‘adb install <path to application>’. Adb is a command line utility which comes as part of the SDK.
OVERVIEW

- BlackBerry Simulators is different from the other emulators as this is Device model specific (rather than OS version)
- Easy to use interface for installing and testing apps

Blackberry Emulators

- Model specific blackberry emulators are available from RIM as standalone window applications and also along with BB JDK.
- The BlackBerry MDS Simulator and the BlackBerry Email Simulator simulates internet and email services respectively.

Prerequisites for Blackberry Simulator

- JRE – Java Runtime Environment
- Blackberry simulator package
- Blackberry email and MDS Simulator

Installing an application on Blackberry emulator

- If the application is in .cod format, it can be directly loaded in to the emulator from the menu options in emulator.
- If the simulator is being used with an IDE or Blackberry JDE it directly loads the application into the simulator and runs it.
- The blackberry device manager can also be used to for installing the applications on emulator and device
Windows Phone 7 is a mobile operating system developed by Microsoft, and is the successor to its Windows Mobile platform. The windows phone 7 emulator comes with SDK and Windows Developer Tool Kit.

Pre-requisites for the emulator

- Windows Vista and higher Desktop OS
- Microsoft Silverlight along with silver light toolkit for windows phone
- Visual Studio 2010
- Windows mobile 7 sdk

Installing applications in emulator

- Open the “Application Deployment” tool which is the part of “Windows Phone Developers Tool”
- Provide the path to the .xap file of the application in the tool
- The emulator opens showing the application that has been installed
Emulators Demo
Limitations

- No support for placing or receiving actual phone calls. You can simulate phone calls (placed and received) through the emulator console, however.
- No support for USB connections
- No support for camera/video capture (input).
- No support for determining connected state
- No support for Bluetooth
- No support for actual GPS
- No support for Accelerometer feature used in Gaming applications
Mobile Testing Tools

Remote Based
- Device Anywhere
- Perfecto Mobile
- Cognizant Mobility Lab

Desktop Based
- Industry Based
  - Squish
  - FoneMonkey
  - JamoSolutions
  - Eggplant
  - SeeTest
- Open Source
  - Robotium
  - Selenium
Tool Selection Criteria

**Scope based decisions**
- Platforms support
- Supported Versions of platforms
- Types of automation supported.
- Device variants within a platforms (form factors)

**Feature based decisions**
- Working on both simulator/device
- Integration to test management
- Ease of use and scripting support

**Infrastructure decisions**
- Stability
- Dependency on Mac/desktops
- Cloud based/stand alone

Cost Benefit Analysis

Timelines for the project

Regression Test Count & Complexity
Some Popular Mobile Testing Tools
Mobile test automation tools are installed on a desktop locally
- Interacts with the mobile devices through USB connection or Wi-Fi
- Mobile Apps can be installed & tested on both Emulators as well as real devices
Most of the Mobile test automation tools work on **Image Based recognition** approach rather than Object based recognition mechanism.

Image based recognition identifies the elements in the application as Images and co-ordinates. Whereas Object based recognition identifies the elements as Object Properties (eg. Text Box 1, Button 2 etc.).

**Pro’s – Image Based:**

- Application technology independent: Does not depend on the application development methodology or the type of mobile application (Native, Hybrid or Web)
- Tools are compatible with all the Mobile OS platforms and versions
- Short learning curve: Usually Image based recognition tools have limited features hence requires shorter learning curve
- Scripts developed for one platform versions can be instantly reused and executed as long as the UI layout does not change

**Con’s – Image Based:**

- Any changes to UI layout will drastically impact the test scripts hence resulting in minimal reusability
- Limited verifications: Cannot verify the object properties like Disabled, Hidden, Drop down list etc.
- Higher long term test script maintenance costs
Mobile Automation – Desktop Based

- **Jamo Solutions** - It extends existing well known environments like HP's Quick Test Professional / Microsoft Visual Studio / Eclipse to create test scripts. The developer can re-use his/her PC-based experience of QTP / Visual Studio/Eclipse to create functional tests and regression tests for mobile applications.

- **Robotium** - Robotium is an open source test framework created to write powerful and robust automatic black-box test cases for Android Native applications easily. With the support of Robotium, test case developers can write function, system and acceptance test scenarios, spanning multiple Android activities.

- **TestQuest CountDown** - TestQuest CountDown is an image comparison and text recognition based tool. It uses an agent to connect to the test devices. The test scripts are GUI based and will be generated by record option.

- **Monkeytalk** – Monkeytalk is an open source functional testing tool specifically for testing iOS and android native applications using record and playback method.

- **Eggplant** - Eggplant is an image-based automation test tool. Eggplant combines image-capture-and-compare technology with a scripting language called 'SenseTalk' that allows professional test teams to automate their repetitive tasks quickly and easily.

- **UIAutomation Framework** - The UI Automation framework comes with Instruments tool in MAC OS. The tool uses component based scripting to perform functional testing. Supports Native and Hybrid application testing on iPhone and iPad.

- **Seetest Studio** - SeeTest allows testing wide variety of applications; Android, iOS, WindowsMobile, Symbian, Blackberry, Windows Phone. SeeTest can be used on both emulators as well as real devices and covers both visual testing and functionality testing. SeeTest plugs into QTP, TestComplete, C#, Java, Python & Perl.
## Mobile Test Tools in Market – A Comparison

<table>
<thead>
<tr>
<th>Tools/ Parameters</th>
<th>Test Quest Countdown</th>
<th>Device Anywhere Pro</th>
<th>Jamo Solutions</th>
<th>Perfecto Mobile</th>
<th>Robotium</th>
<th>Eggplant</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Support of devices/platforms</strong></td>
<td>Android, Blackberry, Symbian &amp; Windows mobile</td>
<td>All major mobile Platforms</td>
<td>Android, Windows Mobile, Blackberry &amp; IOS</td>
<td>All major mobile Platforms</td>
<td>Android</td>
<td>Android, IOS &amp; Blackberry</td>
</tr>
<tr>
<td><strong>Licensing</strong></td>
<td>Cost</td>
<td>Cost</td>
<td>Cost</td>
<td>Cost</td>
<td>OpenSource</td>
<td>cost</td>
</tr>
<tr>
<td><strong>Ease Of Usage</strong></td>
<td>Easy to use</td>
<td>Easy to use</td>
<td>Easy to use</td>
<td>Easy to use</td>
<td>Needs programming</td>
<td>Easy to use</td>
</tr>
<tr>
<td><strong>Scripting</strong></td>
<td>No Scripting</td>
<td>Scripting Possible</td>
<td>Uses VB script</td>
<td>No programming language for scripts, Script is simple and can be easily understood</td>
<td>Uses java</td>
<td>Uses 'Sensetalk'</td>
</tr>
<tr>
<td><strong>Test Data Parameterization</strong></td>
<td>Possible</td>
<td>Possible but complex</td>
<td>Possible</td>
<td>Possible</td>
<td>Possible but complex</td>
<td>Possible but complex</td>
</tr>
<tr>
<td><strong>Remote Connectivity</strong></td>
<td>Possible</td>
<td>Possible</td>
<td>Possible</td>
<td>Not Possible</td>
<td>Not Possible</td>
<td>Possible</td>
</tr>
<tr>
<td><strong>Integration with test management</strong></td>
<td>In-built test management tool, test manager</td>
<td>Quality Center</td>
<td>Quality Center and any other test management tool supported by QTP</td>
<td>Possible</td>
<td>Not Possible</td>
<td>Not Possible</td>
</tr>
<tr>
<td><strong>Jail Breaking / Routing</strong></td>
<td>Needs Jail breaking</td>
<td>DA hosts Jail broken devices</td>
<td>No Jail breaking</td>
<td>Perfecto Mobile hosts Jail broken devices</td>
<td>Needs Jail breaking</td>
<td>Needs Jail breaking</td>
</tr>
</tbody>
</table>
Mobile Testing – Infrastructure Options

Key Considerations for Infrastructure evaluation:

- **Location**: Mobile Testing performed by offshore and onshore teams
- **Emulators / Devices**: Are testing performed using combination of devices / emulators
- **Network modes**: Application needs to be tested on different network modes like Wi-Fi, 3G etc
- **Testing types**: Evaluate different mobility specific testing types
- **Cost**: Consider costs to set up and operationalise lab
1. Freeware and easily downloadable for use
2. It is possible real time scenarios like out of network, Emergency calls etc
3. Since emulator integrates with the development IDE, it would be easy to debug the application for a developer.

1. The Real live interactions cannot be performed (Ex. scanning, capturing etc) - It is not possible to test the applications on a live network connectivity.
2. It just mimics the mobile device from various platforms and hence testing on the emulator cannot guarantee the stability of the application.
3. Some of the interruption test scenarios may also not work properly as like in real handset to predict the actual behavior of the application.
4. Memory Leak issues and Performance issues cannot be detected.
5. Dependency on platform to launch the simulator (Ex. MAC Desktop)
Mobile Infrastructure – Real Devices

1. Testing on actual devices provides access to real device capabilities and constraints. These include device CPU, physical size, manipulation options, screen size, dpi, screen quality, and overall device responsiveness.

2. Represents 100% real user environment testing

3. Testing on devices (using a network SIM card) will also enable you to determine the impact of network speed and latency

4. Ideal case where the target platforms are known and limited

PRO’S

CON’S

1. Testing on actual devices is expensive

2. Continuous investment to upgrade device model

Emulator vs Real devices approach

<table>
<thead>
<tr>
<th>Testing types</th>
<th>Device</th>
<th>Emulator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unit Testing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sanity &amp; Acceptance</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Functionality Testing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interruption</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Regression</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Localization</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Compatibility</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Negative</td>
<td></td>
<td></td>
</tr>
<tr>
<td>User Acceptance Testing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Performance and security</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Legend: Support - No, Support - Yes, Support - Partial
Device selection parameters to be considered

Device Considerations
- Leading Device Manufacturers i.e Samsung, Nokia, Apple, HTC, Motorola
- Types of Devices to be considered i.e Smart phones, Feature phones and Tablets
- Device Capability to be factored i.e Form Factors, device Input methods (Touch / Keyboard), Geo Location based H/W changes

Market Analytics:
- Geo Locations considered
- Popular phones by market region
- Upcoming models to be included
- Types of Phones: (Smart Phones / Feature Phones) popular by market regions

Popular Platforms / Network Considerations
- Android, IOS, BB, Symbian, Windows, Mego
- OS Versions:
  - Minor Update / Major Update
- Network Parameters
  - 2G, 3G, 4G, Wi-Fi and Wi-Max

Configure the selected devices
In the mobile test lab

Selected devices based on assessed parameters configured in the mobile lab

Validate the parameters against Business requirements

Basic lab Set Up
Mobile cloud is a platform enabling testers and developers to easily access multitude of REAL mobile handsets and tablets connected to LIVE mobile networks spread in different geo-locations over Internet.

**Access to latest mobile devices connected to live carrier networks over Internet.**

Dedicated and Public cloud options

Popular vendors include Perfecto Mobile and Device Anywhere

Represents 100% real life as testing is done on a Physical device connected to the actual network

Ideal case for Mobile Cloud includes
Limited testing on variety of devices (like UAT)
Pilot or very short term projects

- Certain types of testing not possible (Bluetooth, Interruptions)
- Licensing cost involved – Typically $8 to $12 per hour usage / device
Mobile Testing Approaches - Infrastructure

**Emulator-based approach:**
- Can be used for performing basic validations.
- Could be used for early validations as well as for system testing needs.
- Cost of the native/browser Emulators are free.
- But the commercial test tools will be of varying cost.

**Device-based approach:**
- Device based approach would be effective for final system testing.
- Effective for testing performance of the applications on actual networks.
- Cost of a device may be around $700 USD approx, and may also need cost of the data plan to be added for network related app testing.
- Commercial test tools will be of varying cost.

**Cloud-based approach:**
- Remote cloud based tools can enable one for testing across a wide array of device families saving on infrastructure cost.
- The vendors like device-anywhere/Perfecto mobile provide this solution.
- Cost would be based on per hour usage and may range from US $10 to $13 on the public cloud and may cost higher for a secured private cloud.

**Combination approach:**
- A right mix of the above could yield a good optimization in terms of coverage as well as the cost.
- Cost of devices, test tools and the cloud based lab would be added for this approach.
Demo
Best Practices
Best Practices for Testing Mobile Applications

- Understand the network landscape and device landscape to identify critical bottlenecks.

- Testing on all target handsets may be the ultimate objective but generally this is impractical if number of target handsets is large, hence Using the Weighted Device Platform Matrix method to identify the most critical hardware/platform combination to test and this will be very useful especially when hardware/platform combinations are high and time to test is low.

- Verifying the end-to-end functional flow in all possible platforms at least once.

- Performance testing, GUI testing, and compatibility testing should be done using actual devices. Even though these tests can be done using emulators, testing with actual devices is recommended.

- Verifying the application behavior with various input/output modalities like target handsets may support one or more of following input modalities like touch, QWERTY keypad, Numeric keypad, Soft keys, 5-Way navigator, and trackball.

- Verifying application behavior against external events like a phone call receive, sms received, a request for a Bluetooth connection, device switched off etc. Make sure that core device operations (phone call, sms) are not affected by the application.

- For applications that use network services, verifying their behavior in different network circumstances i.e. in case of 1) Strong signal strength 2) Intermittent connectivity 3) No connectivity

- Do verify for usability. Importance of usability for mobile applications is even more important. Users who fail to sort out how to use your application or how to access some feature of your application, may simply reject application in favor of some other.

- Start sanity testing with emulators and proceed for further testing on actual devices.
Crash reports are logged automatically on Apple devices. These reports can be accessed by syncing the test device to a computer.

The console log is an iOS feature that includes information from every application on the device. This log can help pinpoint if your app is being adversely affected by other apps/software on the device.

The built-in screen shot command is a good tool to use to document bugs. Holding the home and power button simultaneously will send a snapshot of the device's screen to the iCloud (and subsequently to all your connected devices).

Instruments is an application that traces and profiles iOS code. It is available as part of Xcode Tools. UtilizeLeaks template which monitory memory usage of the app and detects memory leaks.

UIAutomation allows you to perform test automation using Javascript for iOS Native applications. It also provides the ‘Record and Playback’ option.
Few tools and tips that will make Android testing on real devices / Emulators more effective:

- **Change in orientation**
  For devices that support multiple orientations, Android detects a change in orientation when the user turns the device so that the display is "landscape" (long edge is horizontal) instead of "portrait" (long edge is vertical).

- **Change in configuration**
  A situation that is more general than a change in orientation is a change in the device’s configuration, such as a change in the availability of a keyboard or a change in system language.

- **Dependence on external resources**
  If your application depends on network access, SMS, Bluetooth, or GPS, then you should test what happens when the resource or resources are not available.

- **DDMS (Dalvik Debug Monitor Service)**
  Android ships with a debugging tool called the Dalvik Debug Monitor Server (DDMS), which provides port-forwarding services, screen capture on the device, logcat, process, incoming call and SMS spoofing, location data spoofing and more.
Mobile Testing - Process & Methodology

**Requirements Management**
- Involvement of QA in BRD / FRD reviews
- Understand type of mobile app
- Identify Platform versions / Browsers targeted

**Test Strategy**
- Signed off Test Requirements
- Identify the test infrastructure requirements
- Arrive at the test effort using estimation model

**Test Design**
- Prepare Test Scenarios
- Prepare Test Cases
- Define Test Automation framework

**Test Execution**
- Perform test execution on devices / emulators identified
- Manage defects on defect management system

**Test Closure**
- Test Summary report

---

- Requirement traceability matrix to ensure all the requirements are covered during testing
- Mobile test Estimation Model* for estimating efforts for multi platform / multi browser testing
- Strategy for device / emulator execution
- Device Matrix to identify suitable devices to be utilized for testing
- Tool Evaluation matrix to determine the right tool to be deployed for the project
- Platform specific reusable test cases* and UI / Usability checklist
- Capture application logs and screenshots for easier defect triage
- Guidelines for emulator / cloud based / device execution
- Validate against App Store submission guidelines* before app is submitted to market
- Root cause analysis of defects and continuous process improvements

---

* Best Practices:
- Test Estimation Model
- Device Matrix
- Test Design Metrics
- Strategy for device / Emulator
- Tool evaluation Matrix
- Customize UI / Usability checklist
- Platform specific generic test cases
- Reusable test automation assets
- Standards for automation testing
- Test Execution Metrics
- Automation test scripts
- Capture application logs and screenshots
- Defect Metrics
- Guidelines for emulator / device execution

---

Entry Criteria  Exit Criteria
Any offline questions?
Please reach out

Gopinath.Jayamalrao@cognizant.com