WHITEPAPER

Android Multi-Device Challenge

by Guy Arieli, CTO at Experitest

Risk Management in Android Device Testing

there are many Android devices. When doing mobile test automation – say with SeeTest from Experitest - it is not practical to test ALL devices. The question is then - How to select a small number of Android devices that will enable to identify issues across 99% of ALL Android devices?

Summary

In this short article we will suggest two strategies – Edge strategy (testing the most extreme devices) and Commonality strategy (testing the most common, popular devices). Our suggested strategy to obtain optimal results in Android test automation would be a combination of both strategies.

The Challenge – testing thousands of Android devices

The business module behind the Android mobile platform creates huge challenge when you come to test your application. Your application will run on multiple hardware platforms and software versions.
The question is how to set your risk management strategy when you come to select the devices to perform your testing on?
Ideally you would run your entire regression test on all the possible variation of phone model / all Android OS versions / and all vendor firmware versions.
To get an idea of what we are looking at:
As of Q3 2011 there are:

- ~130 (without tablets) different HW modules.
- 7 Android Platform versions: 1.5, 1.6, 2.1, 2.2, 2.3, 3.0 and 3.1.
- The vendor firmware can also vary and can be updated from time to time - a realistic assumption would be ~2 firmware per device.

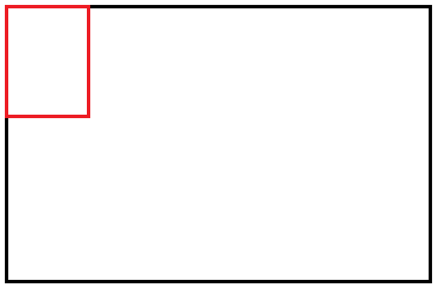
So in order to test all the permutations of Android devices we are looking at around 1800 device combinations (=130 HW device models x 7 SW OS versions x 2 firmware)
For a full list of devices: http://en.wikipedia.org/wiki/List_of_Android_devices

Needless to say this is totally impractical. No mobile automation engineer can seriously target such a large set of devices.

The Solution – identify relevant device factors

The question then is how to narrow down this huge list of thousands of Android devices and select a subset to test on?
The critical thing when narrowing the list is to identify which factors of the device might affect your application behavior and make sure to cover all the possible permutations of such factors. Tackling the factors instead of the devices themselves enables to narrow down to a subset of say 8 devices and provide coverage for ~90% of the devices.
Following is a list of the factors that can affect your application behavior:

- **Screen size** – One of your main concerns is the screen side of the devices that will execute your application. One of the great pitfalls of building an Android application is how to build the view layout so it will render correctly on all screen sizes.
  The screens sizes can vary from QVGA (240x320), WVGA (480x800), HVGA (320x480), FWVGA (480×854) , in tablets you will find 1024x600 and what probably will be the standard for tablets 1280x800.
  So running the same application on 240x320 screens and in the same time on 1280x800 screens is huge challenge.
  To emphasize it please look at the relative size difference (the red rectangular representing the smaller size and the black presenting the larger size):



- **Android OS versions** – The android platform is changing very fast. The difference from version 1.5 to version 2.3 is huge. Lots of new capabilities like graphics HW acceleration were added and can affect your application.
- **CPU** – Mobile devices in general are very sensitive to processing power. We can see phones with single core running at 600 MHz to phones that have duel core 1200 MHz.

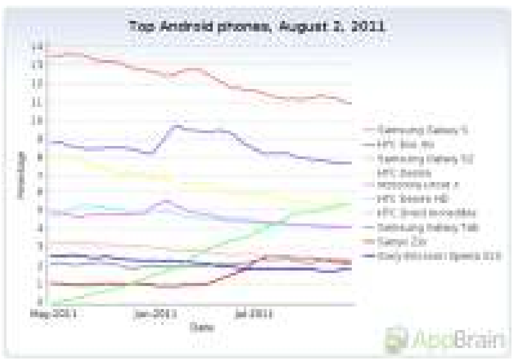
Two other relevant factors of relatively minor effect are GPU (Low-Medium affect) and Manufacture (Low affect).

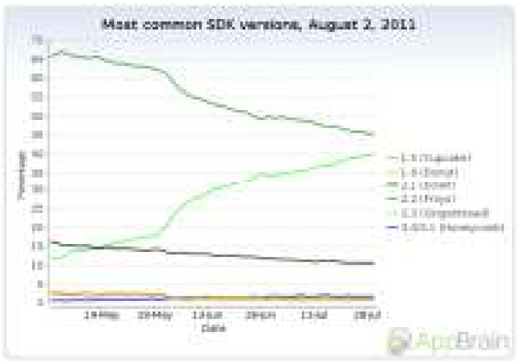
The Strategy – combined Edge & Commonality strategy

To reduce the risk you can work in 2 strategies:

1. **Edge Strategy** - select phones that have factors that are at the edges of the scale: minimum screen size, maximum screen size, running with OS version of 1.5 and running on the latest version and minimum CPU power.
   As we combine them together we can end up with 2 devices:
   On the lower end → HTC Hero: with android 1.5, 320×480 HVGA screen and 528 MHz Qualcomm CPU
   On the highest end → Galaxy Tab 10.1v: with android 3.0, 1280×800 screen and 1000MHz dual core.

2. **Commonality Strategy** – analyze what is the probability each phone has to meet your application and select those with the highest probability.
   The analysis here is more complex and defendant on your application type, region, target age and more.
   There is a huge difference if your application is a business application that is targeted for the US, of if it's a teenager game for girls in Japan.





SOURCE: http://www.appbrain.com/stats/

Our recommendation is to use a combination of the 2 abovementioned strategies, that will probably give you the best result.

About Experitest

Experitest Ltd is the developer of SeeTest – a test automation tool for smartphones including iOS, Android, Blackberry, Windows Mobile (inc. WP7) and Symbian. SeeTest plus into all existing test a automation frameworks such as QTP, C#, RFT, TestComplete, JUnit, Perl, Python.

Download the free trial version from www.experitest.com/download/