



# Real handsets vs. simulators

## COPYRIGHT

---

Copyright © 2011, Perfecto Mobile Ltd. All rights reserved.

The information in this document is subject to change without notice. No part of this document may be reproduced, stored or transmitted in any form or by any means, electronic or mechanical, for any purpose without the express written permission of:

Perfecto Mobile Ltd.

Perfecto Mobile Ltd. assumes no liability for any damages incurred, directly or indirectly, from any errors, omissions or discrepancies between the software and the information contained in this document.

## Contents

<b>1</b>	<b>Introduction .....</b>	<b>3</b>
<b>2</b>	<b>Mobile Simulators .....</b>	<b>4</b>
2.1	History .....	4
2.2	The benefit of using mobile simulator .....	4
2.3	Drawbacks of using mobile simulator .....	5
<b>3</b>	<b>Real handset.....</b>	<b>6</b>
3.1	Background .....	6
3.2	Benefits .....	6
3.3	Drawbacks.....	6
<b>4</b>	<b>Conclusion .....</b>	<b>7</b>

# 1 Introduction

The purpose of this document is to highlight the differences between using simulators and real handsets when testing mobile applications.

A friend working for a leading global carrier once told me “We will start testing on simulators the minute we start selling simulators in stores”. In a nutshell, this represents the major differences between using simulators and real handsets for testing. Eventually, the application is used on real handset and on simulators. The closer you get to the platform, the better quality you will get.

Simulators are very powerful tools for developing mobile applications; they have unique features which enable them to provide a debugging environment. However, by definition, they are simulators and lack the real target environment. Pilots are trained on simulators to test situations that they cannot simulate using real aircrafts as it is obviously a cheaper solution. But can you envision a pilot who never used a real aircraft before to take you across the Atlantic?

## 2 Mobile Simulators

### 2.1 History

At the early days of the simulators, the approach was to develop a PC based application which can mimic the handset by mapping its attributes and behavior. For example, for mobile browsers such attribute is supporting of each HTML element, image types, etc...

As time passes, this approach resulted with two problematic points

1. Devices got complicated, as such; the number of attributes reaches thousands which should be mapped. Just mapping those to every new handset in the market turned out to be an impossible mission.
2. Attribute mapping marked the “positive” side of the handset, but did not emulate its negative sides. Every software component, being PC based or handset based, suffers from flaws, this is the nature of software. As devices got more complicated, this results in the simulators path breaking away from its “home handset” in both unmapping of the handset flaws and the buildup of new flaws only appearing on the simulator itself.

For this reason, you can find such simulators only for relatively low complex handsets. Other, are simply so “far” from the real handset that they have become irrelevant as a testing or even development tool.

New simulators tend to take a different approach and actually compile the handset to work under a normal OS (such as Windows or MAC OS). This approach creates a virtual execution environment which simulates the hardware of the handset and running (as much as possible) the same “binary code” of the handset.

With the first approach you can see simulators coming from Nokia, Openwave

With the second approach you can find simulators coming from Android, Windows Mobile, iPhone and blackberry.

### 2.2 The benefit of using mobile simulator

To start with, it's cheap. In most cases, you will not have to pay anything for using a mobile simulator. You simply need to download it and start using it.

In some cases, mobile simulators might give you the benefit of simulating hard to reproduce scenarios (no network, location, etc...)

Mobile simulators can sometime give you access to detailed information, they might be

integrated with IDE (integrated development environment) which allows you to debug your application step-by-step on the simulator.

## 2.3 Drawbacks of using mobile simulator

Bottom line, you are not testing on your actual platform. This means that even if all goes well, you cannot be sure it will actually work on the handset.

Simple search on the web, will give you hundreds of cases where simulators are having problems not existing on real handsets. These are usually connected to the environment, the sensors, location and especially the network.

The network environment is completely different. In mobile simulators you will use the PC (through any personal firewall), LAN connected to your corporate firewall and to the internet. Using real handsets, the network is connected to the radio interface and from there to the internet. In many cases, the low level network stack itself is completely different. Regardless of whether this is good or bad, it is different, meaning; your application will behave differently.

As opposed to mobile simulators, real handsets have limited resources. The mobile handset is a compact platform with different CPU power and memory (this can be extended to other related "stuff" such as graphic chipset, input devices, etc...). What's running on a powerful quad-core intel based CPU with 4GB of memory might not work properly (if at all) on single core, limited CPU with 500M of memory.

When addressing testing using simulators, a tester might stumble a question of "It doesn't work – what now?". One can envision several possible scenarios.

1. It doesn't work on the simulator, need to test it on real handset
2. It probably doesn't work at all

Obviously, yet another "hidden" scenario is "it does work on the simulator, what's now?" – is this sufficient or should I also test this on real handset.

## 3 Real handset

### 3.1 Background

When considering the area of testing on real handsets, there are several alternatives to consider. By definition, handsets are physical resources which need to be managed. Unlike simulators where additional “handsets” can be either additional software installed or simple configuration, real handsets are something physical to own.

Services such as Perfecto Mobile allow you to virtualize those resources; however, those are still eventually real physical resources.

Given this important fact, individuals testing on real handsets always had to choose how to manage those resources. In many cases, those have become “rare” resources for testers who had to define their priority list based on platform, brand, screen size, OS, etc...

Beside “side projects” or “garage projects”, testing on real handsets is not an optional task, it is always a given task, and the only question to ask is to which extent.

### 3.2 Benefits

Testing on real handsets always give you the actual results. There is no need to worry about “false negatives” and more important there is no need to worry about “false positive” situations. The later, is crucial, as those are scenarios which have passed the testing on simulators but will fail on real handsets.

Real handset testing will give actual results. Those will naturally take into account the network, CPU, memory, screen size, etc... The nice thing about the environment is that there is little to do, for good and bad, the real handset is the real handset.

It is much easier to expose performance defects with real handsets as well as defects which are the results of the handset itself or its environment. All of those are, normally, not detected using simulators.

### 3.3 Drawbacks

As stated above, unless somehow managed, handsets are physical resources which need to be managed. Both in hardware and in cellular plans, networking, definition, etc.

Real handsets are harder to connect to IDE. Although at later phase, this requirement is less relevant (important for early stage development), it is still a missing functionality.

## 4 Conclusion

The goal of testing is to make sure everything is working as expected. When using real handsets, the situation is quite simple, if it works, it works, if it doesn't, it needs to be fixed.

Simulators are very good for the early days of application development; their integration with the development environment IDE is very powerful. On the other side, before releasing the application (or web site) into the market, you probably do want to test it on real handsets (as it will probably be very hard to blame the simulator).

The “gray” area between those two edges might vary between different organizations and needs. In most cases, the rule of thumb would be that actual development should use simulators (and few reference real handsets) and testing should be done on real handsets.

If you are a “garage” developer who is willing to take great risks with your application, simulators are good and very cost effective solution for you. However, if you are developing an application for your organization for internal or external use, you should probably use real handsets for your testing as you don't want those “flaws” to be first found in production mode.

Use the below “acceptable equation” to do the math yourself, based on your needs

Found defect cost in R&D = 10% of found defect cost in QA = 1% of found detect cost in production.

The above equation has better statistical clearance with clear market applications. If the value of the application is clear, the equation is true. In other cases, the value is not that clear.

References

A guide to mobile emulators

<http://mobiforge.com/testing/story/a-guide-mobile-emulators>

Nokia – Mobile device emulator

[http://library.forum.nokia.com/index.jsp?topic=/Java\\_Developers\\_Library/GUID-7166F002-E919-4CD7-9FAF-53F6B18207F6.html](http://library.forum.nokia.com/index.jsp?topic=/Java_Developers_Library/GUID-7166F002-E919-4CD7-9FAF-53F6B18207F6.html)