University of Strathclyde

Department of Bioengineering

Smart phones as medical devices

By

Luis Javier Rico Arias

A thesis presented in fulfilment of the requirements for the degree of Master Science in Bioengineering

2011

DISCLAIMER

Signed:

Date:

Acknowledgements

I would like to thank my supervisor Dr Bernard Conway for the support received and the sharing of his knowledge and experience during the process and writing of this thesis.

Abstract

Of the 1.5 billion mobile phones sold worldwide in 2010 more than 296 million were smart phones[1]. This accessibility matched with powerful and fast processors, memory capacity, sensors and communications amongst other features mean smart phones could be used as medical devices. This project aims to explore this possibility and scope an application based on elements of smart phone capability.

The ability to communicate with other devices means that gathered, processed and stored data could be sent on or used appropriately to prevent, treat or monitor illness.

To report on cost, type of device, ease of use and availability, a systematic search for applications (APPs) that use smart phones as medical devices was carried out focussing on the major market.

A background review of the operating systems (OS) available on the current market was also conducted in order to inform on the practical elements of the project. The review was focused on the four major OS providers and the other most significant smart phone OS providers. The major OS are Android OS, iOS, Blackberry OS and Windows Phone 7 OS[2]. The OS review examined price, fees, simulator availability and the ease of developing and testing code on the chosen device via different software development kits (SDK). A comparative evaluation of each OS was performed and coupled with a summary of market trends to establish which devices provide the most popular and effective future platforms for medical device application developments. Central to the review was the comparison of ease of access to sensor data (in-built and external), the communication protocols, the ports, processing speed, data security, memory, and battery life.

Based on this review a recommendation of a platform smart phone was identified for demonstration purposes and a simple application capturing, processing and reporting on a sensed measurement was performed.

Contents

Tables

Figures

1. Introduction

Over 296 million smart phones were sold worldwide during 2010 and there is an expected increase of 57.7% in sales for 2011[3]. This increase in sales and therefore in the number of devices that the public own, make them a very interesting field to explore with new and interesting ways of patient monitoring and diagnosis. Smart phones are becoming very powerful devices; the latest releases in the market have dual core processors working at frequencies of up to 1.2GHz and have powerful graphic capabilities. The sensors that are packed into smart phones together with their communication and storage capabilities mean that a patient could be monitored 24 hours a day 7 days a week anywhere in the world and with internet access this monitoring could be done in real time.

There are over 225,000 android applications on the android market with between 7,000 and over 22,500 applications being added per month within the past 12 months[4]. There are over 500,000 applications in the app store for iOS, the iphone and iPad operating system[5]. There are over 20,000 applications for Windows Phone 7 operating system and over 38,000 applications can be found in the BlackBerry App World. Amongst those applications health monitoring and diagnosis applications can be found. To get an estimate of the number of applications related to health, the market was researched. As a result of this research an overview of prices and efficiency of applications was established and gaps in the market were identified.

A review of the operating systems (OS) available in the current market was conducted. The review focussed on Android OS and iOS. The OS review examined the price, fees, simulator availability and the ease of developing and testing code on the chosen device via different software development kits (SDK). A comparative evaluation of each OS was performed and was coupled with a summary of market trends to establish which OS provide the most popular and effective platform for medical device application developments.

A review of the devices available on the current market was conducted. The evaluation took into account the type of screen, processor speed, graphics processing unit, number of available sensors, communications connectivity, price, operating system, RAM, internal memory, possibility of expanding the memory and capacity of the battery.

A discussion of the recommended device based on current technologies, OS and type of application to be developed is given under the platform recommendation section. In that section a discussion about the different applications found and the OS and devices reviewed is performed together with a rationale for final recommendations.

A look into the medical device directive to check the regulatory hurdles in using smart phones in the clinical environment is performed and discussed under the medical device directive section.

In the practical section of the thesis software development and programs used to begin the development of an application is provided to illustrate coding and simulation of app elements together with screen shots.

The specifications of a potential application which can make use of the in-built special features of the smart phone human machine interface is then given along with potential problems needed to be overcome before such an application would be operational.

The thesis concludes with a discussion on the challenges and opportunities associated with implementation of applications and simulations of medical devices based on smart phone technology. The recommendations, viability and future direction of this type of work is also covered in these sections.

2. Literature review

The aims and objectives of this section were:

- Research the different markets and stores seeking medical applications.
- Review the different Operating systems
- Analyse the current market of available smart phones

2.1. Medical applications by operating system

The medical applications found within the different application libraries, stores or markets referred mostly to medical texts, dieting, exercising and inserting data to keep track of different pathologies such as glucose levels in blood and heart rate. Associated with this are services such as Microsoft's HealthVault and Google Health that store the patient's data in the network and can be accessed from anywhere providing there is internet access. In addition to these, some applications have their own storage networks to store the information so a clinician can access it from the office or hospital. However, providing data repositories to the general public in which they store personal health and wellbeing data may not be sustainable as Google recently announced that it intends to end its Google Health product early in 2013 due to lack of uptake by consumers. It may be that the public remain sensitive to the idea of health data being held by non-government organisations.

There are many applications that relate to health/medical, too many to list them all in tables. The focus was on finding applications that use internal or external sensors. The applications that were not included in the research were the ones dedicated to giving medical or health advice, medical encyclopaedias, symptom finding, record keeping and management and those that did not use Bluetooth or other methods of communication to connect to the phone to transfer information gathered.

In the following sections, tables with the focus on medical applications have been produced. The following tables indicate the amount of memory used by each of the applications in megabytes (MB), if the applications use internal or external sensors, communicate to other devices and if information is stored internally or in a network. An analysis of each table is shown at the end of each section where the most common type of application is identified, prices are studied and the prices of some external sensors necessary for some applications are shown.

Medical applications for Android

App name	AirStrip OB	AlertHey Health Monitoring	Bluetooth Blood Glucose	Bluetooth Pulse Oximeter	btHeart-Workout monitor	Cardio Trainer	Glucose Meter	Heart rate monitor
Price (£)	0	0	0.61	0.61	0.62	0	1.71	0
Size MB	0.9	1	2.8	3.3	0.67	6.4	1.44	0.1
Speaker Earphones	X	X	X	X	X	X	X	X
Microphone	X	X	X	X	X	X	X	X
Camera	X	X	X	X	X	X	X	X
Flash	X	X	X	X	X	X	X	X
Accelerometer	X	X	X	X	X	✓	X	X
Bluetooth	X	✓	✓	✓	✓	✓	✓	✓
GPS	X	X	X	X	X	✓	X	✓
External device	✓	✓	✓	✓	✓	✓	✓	✓
Sotre data on phone	X	X	✓	✓	X	✓	✓	X
Store data on server	X	✓	X	X	X	X	✓	X
Remote monitoring	✓	✓	X	X	X	X	✓	X

App name	Heart rate monitor	Instant heart rate	Pulse rate monitor beta	Response Measurem	Zephyr MD	Zephyr Me	Tinnitus Hyperacusis suppressor PRO	Test-Your-Hearing
Price (£)	0	0	0	0	0	0	0.88	0
Size MB	0.28	1.65	0.055	1.04	0.61	1.02	2.8	11
Speaker Earphones	X	X	X	X	X	X	✓	X
Microphone	X	X	X	X	X	X	X	✓
Camera	X	✓	✓	X	X	X	X	X
Flash	X	X	X	X	X	X	X	X
Accelerometer	X	X	X	✓	X	X	X	X
Bluetooth	✓	X	X	X	✓	✓	X	X
GPS	X	X	X	X	X	X	X	X
External device	✓	X	X	X	✓	✓	X	X
Sotre data on phone	X	X	X	X	X	✓	X	X
Store data on server	X	X	X	X	✓	✓	X	X
Remote monitoring	X	X	X	X	✓	✓	X	X

Table 1, Android applications

Table 1 provides a review of applications associated with the Android OS. As can be seen in table 1 most of the applications found on the android market that use an internal or external sensor are related to heart monitoring or blood oxygenation. Although there are some applications that relate to blood sugar monitoring, there are not as many as for heart monitoring. It can also be seen that most of the applications are free. However, most of the free applications required the use of external devices. These devices are the sensors and the phone is used as a display, to store data on the device or send the data to a server where it can be accessed by authorised users like clinicians, patients, carers or family. When there is a fee for a software application it is noticeable that they are not costly, £1.71 being the most expensive. However, cost escalates when external options are necessary. The external sensors used with the ZephyrMe application for example use a BioHarnes that is situated around the chest of the individual to be monitored. The BioHarnes monitors the heart rate, breathing rate, IR skin temperature, position/posture and activity in VMU. It has a 3 axis accelerometer, a skin conductance level, a multicoloured indicator showing subjects status, 250Hz ECG logging, 125Hz accelerometer logging, USB connectivity for data download, Bluetooth connectivity and 570 hours data storage[6]. The price of the harness is $710.00 and it stated to be usable for team and individual fitness training, first responder monitoring, E-Health (remote monitoring of patients vital sign data) academic research and consumer fitness applications (using smart phones and web portals) as stated in the web site[7].

2.1.1. Medical applications for iOS

App name	BAM Labs	BeatMonitor	Heart EKG	Heart R8	Heart Rate	Instant heart rate	iStethoscope Pro	Pulse Phone
Price (£)	0	1.19	1.79	0	0	0.59	0.59	0.59
Size MB	1.1	0.9	1.1	1.1	1.1	4.7	0.5	0.4
Screen	X	X	X	X	X	X	X	X
Speaker Earphones	X	X	X	X	X	X	X	X
Microphone	X	✓	✓	X	X	X	✓	X
Camera	X	X	X	✓	✓	✓	X	✓
Flash	X	X	X	X	X	X	X	X
Accelerometer	X	X	X	X	X	X	X	X
Bluetooth	X	X	X	X	X	X	X	X
GPS	✓	X	X	X	X	X	X	X
External device	✓	X	X	X	X	X	X	X
Sotre data on phone	X	✓	✓	✓	✓	✓	✓	✓
Store data on server	✓	X	X	X	X	X	X	X
Remote monitoring	✓	X	X	X	X	X	X	X

App name	Pulse rate	Pulse-o-matic	Skin prevention	Stressometer	Tremor tracer	Visual Heart Rate monitor	Visual Heart Rate monitor	iPulseRate
Price (£)	0.59	0.59	6.99	0.59	2.39	0	1.19	1.19
Size MB	0.1	5.8	4.3	0.7	0.2	2.7	2.2	0.5
Screen	X	X	X	X	✓	X	X	X
Speaker Earphones	X	X	X	X	X	X	X	X
Microphone	✓	✓	X	X	X	X	X	X
Camera	X	✓	✓	X	X	✓	✓	✓
Flash	X	X	X	X	X	X	X	✓
Accelerometer	X	X	X	✓	X	X	X	X
Bluetooth	X	X	X	X	X	X	X	X
GPS	X	X	X	X	X	X	X	X
External device	X	X	X	X	X	X	X	X
Sotre data on phone	✓	✓	✓	✓	X	✓	✓	✓
Store data on server	X	X	X	X	X	X	X	X
Remote monitoring	X	X	X	X	✓	X	X	X

App name	iHeart-Pulse Reader	SporTrackr	Motor	RVG Mobile	Mobile MIM	DermScope	uHear	Hearing Check
Price (£)	7.79	2.99	0	0	0	2.99	0	0
Size MB	1.8	5.4	0.1	9.5	6.3	7.8	5.3	1.9
Screen	X	X	X	X	X	X	X	X
Speaker Earphones	X	X	X	X	X	X	✓	✓
Microphone	X	X	X	X	X	X	X	X
Camera	X	X	X	X	X	✓	X	X
Flash	X	X	X	X	X	✓	X	X
Accelerometer	✓	✓	✓	X	X	X	X	X
Bluetooth	X	X	✓	X	X	X	X	X
GPS	X	✓	X	X	X	X	X	X
External device	X	✓	X	✓	✓	X	X	X
Sotre data on phone	✓	✓	✓	✓	✓	✓	X	X
Store data on server	X	X	✓	X	✓	X	X	X
Remote monitoring	X	X	✓	✓	✓	X	X	X

App name	Hearing Test	iHear Hearing Aid	Hearing test #1	soundAMP Lite	soundAMP R	iCheck Hearing	Audiometry	iTinnitus Solutions
Price (£)	0.59	0.59	1.19	0.59	2.99	0.59	1.79	2.99
Size MB	12.4	0.3	0.3	6.6	6.6	0.8	0.8	5.1
Screen	X	X	X	X	X	X	X	X
Speaker Earphones	✓	✓	✓	✓	✓	✓	✓	✓
Microphone	X	✓	X	✓	✓	X	X	X
Camera	X	X	X	X	X	X	X	X
Flash	X	X	X	X	X	X	X	X
Accelerometer	X	X	X	X	X	X	X	X
Bluetooth	X	X	X	X	X	X	X	X
GPS	X	X	X	X	X	X	X	X
External device	X	X	X	X	X	X	X	X
Sotre data on phone	X	X	X	X	X	✓	✓	✓
Store data on server	X	X	X	X	X	X	X	X
Remote monitoring	X	X	X	X	X	X	X	X

App name	iHearClearly	Amplify'd	Senses - What U Hear	Hearing Test4All	Ringing Relief Pro	Tinnitus Relief Pro	EARs	WiScale
Price (£)	1.19	0.59	0	0.59	1.79	2.99	2.39	0
Size MB	0.1	4.3	33.9	0.8	7	13.7	0.3	3
Screen	X	X	X	X	X	X	X	X
Speaker Earphones	✓	✓	✓	✓	✓	✓	✓	X
Microphone	✓	✓	X	✓	X	X	✓	X
Camera	X	X	X	X	X	X	X	X
Flash	X	X	X	X	X	X	X	X
Accelerometer	X	X	X	X	X	X	X	X
Bluetooth	X	X	X	X	X	X	X	X
GPS	X	X	X	X	X	X	X	X
External device	X	X	X	X	X	X	X	✓
Sotre data on phone	X	X	X	✓	X	X	X	✓
Store data on server	X	X	X	X	X	X	X	✓
Remote monitoring	X	X	X	X	X	X	X	✓

Table 2, Applications for iOS

As can be seen on table 2, there are more hearing related applications than heart related applications, also it can be seen that the applications are more reliant on the sensors provided by the phone. Applications that use external sensors are also available like WiScale which can use a Wi-Fi body scale to measure the weight and a cuff to check the blood pressure and heart rate, both of them manufactured and sold by Withings. The WiScale application is free but the scale and the blood pressure monitor costs **€** 129.00 each[8].

The applications come in a variety of prices from free up to £7.79. Most of the priced applications do not required an external device while some of the free applications do required an external device to work as intended.

The dual microphone of the iPhone gives the possibility to create applications that can utilise noise cancelation. In this context, example applications use the phone as a stethoscope or applications that amplify sound.

The hearing related applications use the phone to test the hearing of the patients, to amplify sounds sensed by the microphone and outputs them through the speaker or through earphones. A number of applications aim to relieve people suffering from tinnitus by playing sounds through earphones.

2.1.2. Medical applications for Blackberry OS

App Name	Price ($)	Size (MB)	microphone earphones	Microphone	Camera	Flash	Stores data on phone
White Noise	0.99	0.288	✓	X	X	X	X
Hearing and Eyesight test	2.99	1.36	✓	X	✓	X	✓

Table 3, Applications for BlackBerry

As can be seen in table 3 there were not many relevant applications in the BlackBerry OS market. The two applications that were identified use internal sensors and were related to hearing and eyesight. These applications cost $0.99 and $2.99 respectively.

No other relevant applications were identified in the BlackBerry market which used any internal or external sensors. There were many applications to log and track blood sugar, heart rate and blood pressure and medical reference books.

2.1.3. Medical applications for Windows Mobile 7

No medical applications were found in the Windows Mobile 7 market place.

2.2. Operating Systems

Android OS, iOS, BlackBerry OS and Windows Mobile 7 OS are analysed in the following sections. The following are considered in this section; the requirements needed to develop applications within the different operating systems, the programming language that is used, resources provided by the different companies, any payable fees, the requirements needed to upload the applications into the phones for testing, requirements needed to launch the application into the market and market trends of the operating system.

2.2.1. Android OS

Android operating system is used to run mobile devices like smart phones, tablets, laptops and televisions. Android Inc was started by Andy Rubin, Rich Miner, Nick Sears and Chris White in 2003[9], it was sold to Google Inc in 2005.

Android OS is based on the Linux kernel and most of the code was released under the Apache License, a free software license and open source license[9]. The software and everything needed to develop applications is free, the software is open source.

The operating system consists of Java applications. Figure 1, Android architecture shows that the final applications lie on top of the applications framework used for managing applications. Below this level there is the Android runtime libraries and Dalvik virtual machine and the libraries, which use C and C++ languages can be found just on top of the Linux Kernel[10].

Android has the capacity of multitasking and each application uses a different instance of the Dalvik virtual machine in its own process. The threading and low level memory management are underlying functionalities that are possible due to the Linux kernel and access through the Dalvik virtual machine[10].

The software needed to start developing Android applications is:

- Java Development kit[11]
- Eclipse IDE[11]
- Android software development kit[11]
- ADT Plug-in for Eclipse[10]

The programming is all done in Eclipse IDE where different files are created automatically when first starting a project. The Android SDK is installed within Eclipse IDE as well as the ADT plug in. All this software is available free of charge on the internet from different websites.

Installation procedures and all the information needed to program applications can be found on android.com under the developers section. Within the developers section the SDK, the developers guide, reference, resources, videos and a blog can be found. In those sections the different parts of how to get started, develop an application, the libraries with their key words and examples of how to use them and a blog where developers can ask for advice can be found.

A simulator is included within eclipse where all the programming can be tested before downloading into the phone.

The program can be tested at any stage directly on the phone and when finished it can be uploaded onto the Android market place or it can be made available on any other website. If the program is made available through the market place a Google account is needed and a registration fee of $ 25.00 has to be paid as well as agreeing to the Android Market Developer Distribution Agreement. The application can be available for free or a price can be set. If the application has been made available through the Android market and is free of charge to the users Android does not charge for its distribution. If the final users have to pay to download the application there is a transaction fee of 30% of the application price. However if the application is sold in any other web server the fees will be those requested by that server.

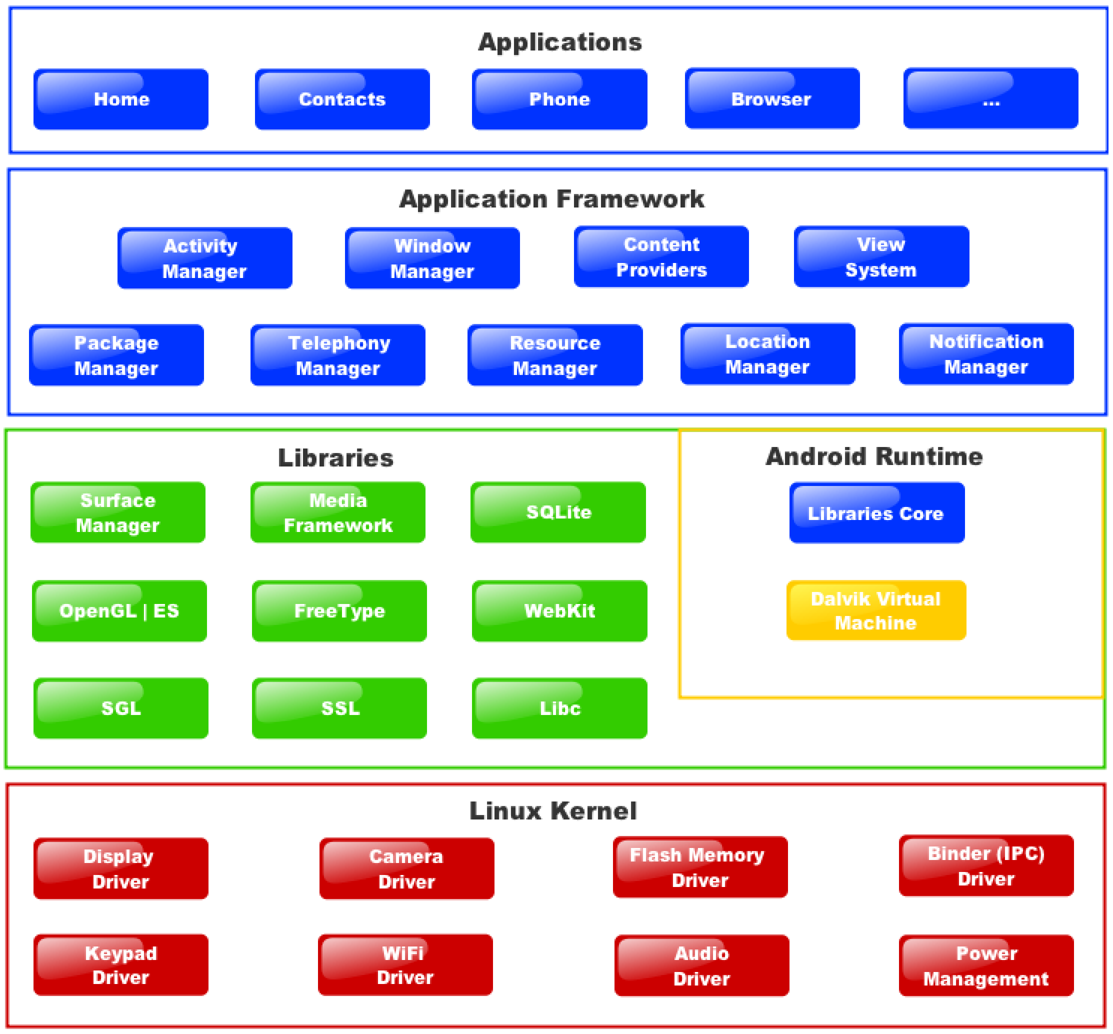
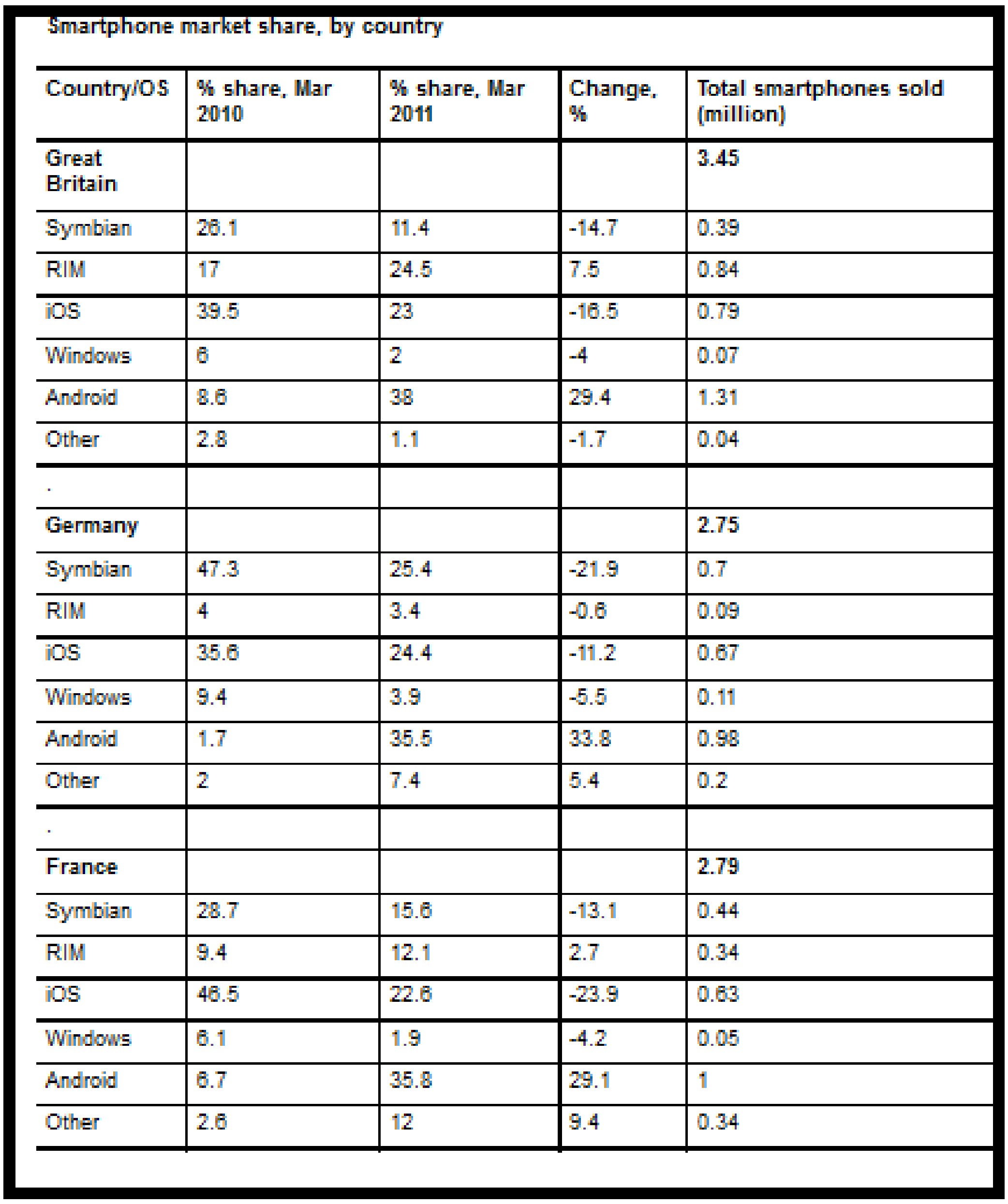


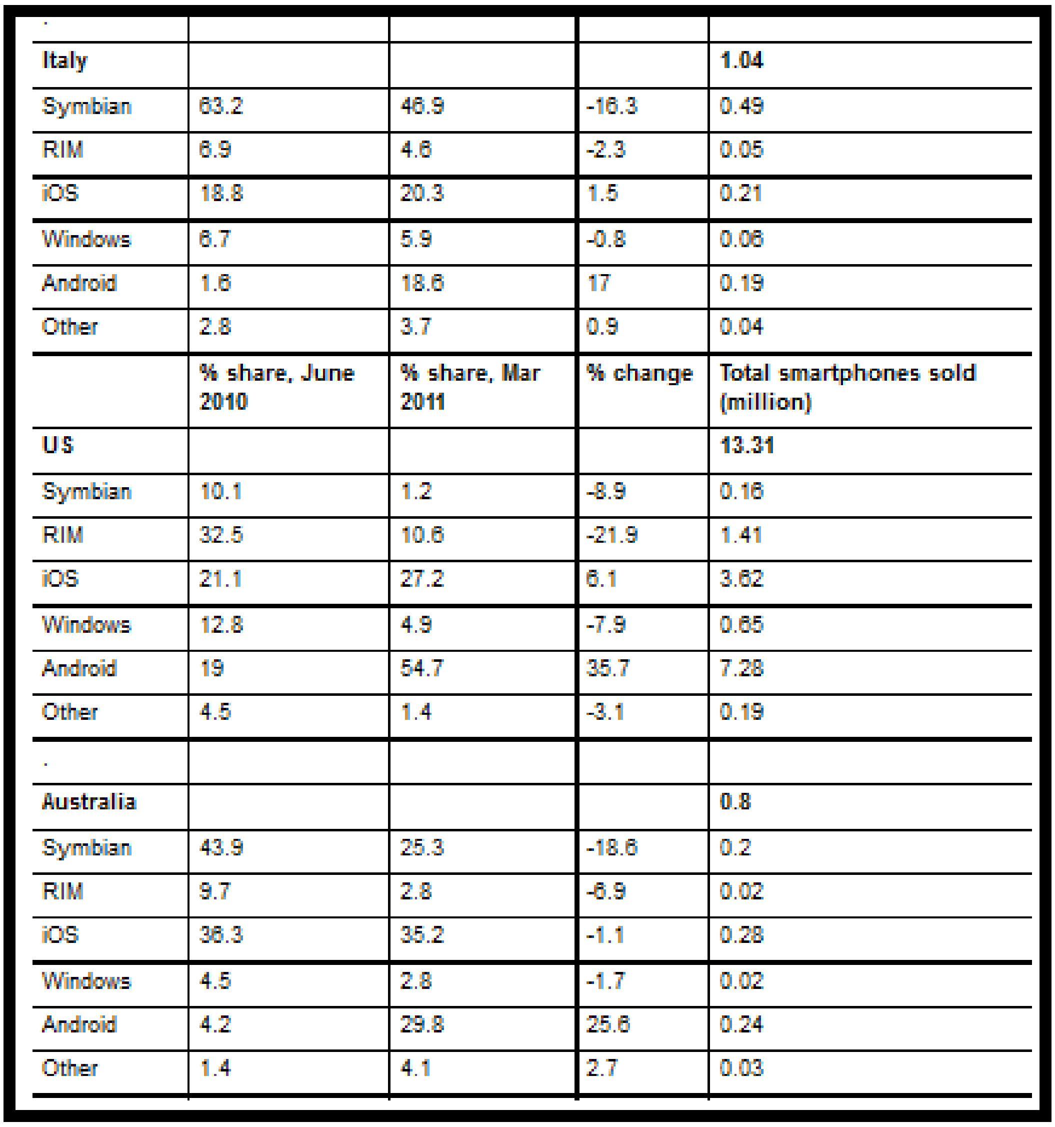
Figure 1, Android Architecture[9]



Smartphone market share, by country

Country/OS	% share, Mar 2010	% share, Mar 2011	Change, %	Total smartphones sold (million)
Great Britain				3.45
Symbian	26.1	11.4	-14.7	0.39
RIM	17	24.5	7.5	0.84
iOS	39.5	23	-16.5	0.79
Windows	6	2	-4	0.07
Android	8.6	38	29.4	1.31
Other	2.8	1.1	-1.7	0.04
.				
Germany				2.75
Symbian	47.3	25.4	-21.9	0.7
RIM	4	3.4	-0.6	0.09
iOS	35.6	24.4	-11.2	0.67
Windows	9.4	3.9	-5.5	0.11
Android	1.7	35.5	33.8	0.98
Other	2	7.4	5.4	0.2
.				
France				2.79
Symbian	28.7	15.6	-13.1	0.44
RIM	9.4	12.1	2.7	0.34
iOS	46.5	22.6	-23.9	0.63
Windows	6.1	1.9	-4.2	0.05
Android	6.7	35.8	29.1	1
Other	2.6	12	9.4	0.34

Figure 2, 2011 Smartphone market share, by country [12]



.				
Italy				**1.04**
Symbian	63.2	46.9	-16.3	0.49
RIM	6.9	4.6	-2.3	0.05
iOS	18.8	20.3	1.5	0.21
Windows	6.7	5.9	-0.8	0.06
Android	1.6	18.6	17	0.19
Other	2.8	3.7	0.9	0.04
	% share, June 2010	**% share, Mar 2011**	**% change**	**Total smartphones sold (million)**
US				**13.31**
Symbian	10.1	1.2	-8.9	0.16
RIM	32.5	10.6	-21.9	1.41
iOS	21.1	27.2	6.1	3.62
Windows	12.8	4.9	-7.9	0.65
Android	19	54.7	35.7	7.28
Other	4.5	1.4	-3.1	0.19
.				
Australia				**0.8**
Symbian	43.9	25.3	-18.6	0.2
RIM	9.7	2.8	-6.9	0.02
iOS	36.3	35.2	-1.1	0.28
Windows	4.5	2.8	-1.7	0.02
Android	4.2	29.8	25.6	0.24
Other	1.4	4.1	2.7	0.03

Figure 3, 2011 Smartphone market share, by country[12]

OS	2010	2011	2012	2015
Symbian	111,577	89,930	32,666	661
Market Share (%)	37.6	19.2	5.2	0.1
Android	67,225	179,873	310,088	539,318
Market Share (%)	22.7	38.5	49.2	48.8
Research In Motion	47,452	62,600	79,335	122,864
Market Share (%)	16.0	13.4	12.6	11.1
iOS	46,598	90,560	118,848	189,924
Market Share (%)	15.7	19.4	18.9	17.2
Microsoft	12,378	26,346	68,156	215,998
Market Share (%)	4.2	5.6	10.8	19.5
Other Operating Systems	11,417.4	18,392.3	21,383.7	36,133.9
Market Share (%)	3.8	3.9	3.4	3.3
Total Market	**296,647**	**467,701**	**630,476**	**1,104,898**

Figure 4, Gartner Worldwide OS market share (number of devices in thousands of units)

As can be seen in figures 2 and 3 Android OS leads the market in Great Britain, Germany, France and US. It is the second most popular operating system in Australia and the third most popular OS in Italy. It also can be seen that Android OS has grown a considerable amount from the year 2010 to 2011 in all countries. The country with lowest growth was Italy with a 17% change from one year to the next. The country where Android has grown the most was the US with a 35.7% growth.

Table 4 shows the market share of the different operating systems available since 2010 and gives a prediction of the market until 2015. A considerable growth of the Android OS platform can be seen, a less steep growth for Microsoft followed by just 1.5% growth by Apple's operating system. The market shares for the rest of the operating systems shrink showing the biggest losses on Symbian follow by Research In Motion.

From those 3 tables it can be assumed that the most popular operating systems in the current market is Android OS.

2.2.2. iOS

iOS was developed by Apple for the iPhone and both were released in June 2007, this operating system is now only available for the iPod, iPad, iPhone and Apple TV. It cannot be run in third party devices. iOS is based on Apple's Mac operating system X and therefore has the Darwin foundation and Unix OS basis in common. Both Android and iOS can be subdivided into the Core OS layer (Android is the kernel), the core services layer (in Android are the libraries), the media layer (Applications framework for Android) and the Cocoa Touch layer (the applications in Android). Third party applications could not be run until March 2008 when the first beta version of the Software Development Kit (SDK) was released. Objective-C, C and C++ is used to program applications as for Mac OS X[13].

To start programming iOS applications, an Apple computer is needed as well as one of the developers programs shown in table 4. The programs give all the documentation and the SDK to start to develop the applications. A higher education version is available to universities but the university program does not offer the Code-level technical support, Ad Hoc Distribution and the App Store Distribution.

Applications cannot be tested on the iPhone unless one of the previously mentioned programs has been granted by the company. The distribution of the applications has to be done through the iTunes app store. The application can be distributed for free, and then no fees have to be paid. If a price is set for an application then Apple charges 30% of the price.

As shown on figures 2 and 3 the market share for iOS between 2010 and 2011 for the UK, Germany, France and Australia has shrunk. The market share in the UK has dropped by 16.5%, in Germany by 11.2%, in France by 23.9% and in Australia by just 1.1%. For US and Italy the market share for iOS has increased by 6.1% and 1.5% respectively.

Worldwide market share projections from Gartner shows that iOS had a market share in 2010 of 15.7% of the market, and says that has increased to a peak of 19.4% in 2011 but that for 2012 and 2015 the market share for iOS will be reduced to 18.9% the first year and 17.2% on 2015.

	University Program For higher education institutions looking to introduce iOS app development into their curriculum.	Standard Program For developers primarily creating free and paid iOS apps for distribution in the App Store. Learn more	Enterprise Program For companies and organizations who will be creating proprietary iOS apps for internal distribution. Learn more
Pre-release versions of iOS SDK	N/A	✓	✓
GM versions of iOS SDK	✓	✓	✓
iOS Dev Center Resources	✓	✓	✓
Apple Developer Forums	✓	✓	✓
Test on iPad, iPhone, and iPod touch	✓	✓	✓
Code-level technical support	N/A	✓	✓
Ad Hoc Distribution	N/A	✓	✓
App Store Distribution	N/A	✓	N/A
In-house Distribution	N/A	N/A	✓
Cost	Free	$99 USD/year	$299 USD/year

Table 4, iOS developers programs [14]

2.2.3. BlackBerry OS

BlackBerry offers different free software development kits to develop applications in different languages. The language used is C++ but it is possible to use Eclipse and do the programming in Java language.

A registration is necessary to start developing and to be able to download different software packages from the BlackBerry website.

To publish the applications the developer must apply to be registered and pay a registration fee. Once accepted the applications can be sent to BlackBerry where they are analysed. If the application is approved it will be published on the BlackBerry app world. There is a payable fee for each application that is sent to BlackBerry for publishing.

The market share of BlackBerry has fallen in almost all of the countries shown in figures 2 and 3 apart from in Great Britain and France where it has increased. The world market share predictions made by Gartner, shows in figure 4 that Research In Motion (RIM) market share is going to decrease steadily to reach just 11.1% by 2015.

2.2.4. Windows Phone 7

Windows Phone 7 operating system development started in 2008 by Microsoft and was first released in Europe and other parts of the world in October 2010. Windows Phone 7 uses a new user interface which has been developed using Metro, a new design language [15]. Applications that have been developed for Windows Mobile OS are not compatible with Windows Phone 7.

The software needed to develop the applications is available at the Microsoft website. The programs that are necessary to develop the applications are:

- Microsoft Visual Studio 2008 Professional Edition or Microsoft Visual Studio 2005 Standard Edition or above ($779.00)
- Windows Mobile 6.5.3 developer's tool kit (DTK) (Free)
- Microsoft.NET Compact Framework v2 SP2 (Free)
- Windows Mobile Center (Free)

There is a free Express version of the Microsoft Visual Studio but it is not supported by Windows Server 2003, XP, Vista, Server 2008 and 7 [16].

Market share for Microsoft has decreased in all of the countries shown in figures 2 and 3. The biggest fall has been in the US with a 7.9% market share loss and Italy has been the place with the least market share decreases to just 0.7%. The world market share prediction from Gartner estimates that Microsoft will grow from 4.2% in 2010 to a 19.5% in 2015 as shown in figure 4. During the first quarter of 2011 alone Windows Phone 7 has reached 1.6% of the worldwide market share [15].

2.3. Smart phones devices

On the following tables, 34 of the most recently released and soon to be available (summer 2011) smart phone devices can be found

Mobile Phone	**BlackBerry Storm**	**BlackBerry Torch**	**Casio G'zOne Commando**	**Sony Xperia arc**	**Sony Xperia Play**	**Sony Xperia pro**	**Sony Xperia X10**
Price (£)	289.99	344.99	X	444.99	439.99	489.99 due	276.99
Operating systems	BlackBerry OS 5.0.0.419	BlackBerry OS 6.0	Android 2.2	Android 2.3	Android 2.3	Android 2.3	Android 1.6
Processor	528 MHz Qualcommprocessor	624 MHz Marvell Tavor PXA940 [2] (ARM Cortex A8) compliant	X	Qualcomm Snapdragon MSM8255	Qualcomm Snapdragon MSM8655	Qualcomm Snapdragon MSM8255	Qualcomm Snapdragon QSD8250
Processor speed (GHz)	0.528	0.624	0.8	1	1	1.024	1
GPU	X	X	Adreno 220	Adreno 205	Adreno 205	Adreno 205	Adreno 200
Touch screen	Capacitive	Capacitive	Capacitive	Capacitive	Capacitive	Capacitive	Capacitive
Screen size (inch)	3.25	3.2	3.6	4.2	4	3.7	4
RAM	128 MB	512 MB	512 MiB	512 MiB	512 MiB	512 MiB	384 MiB
Memory	1 GB	4 GB	X	513 MiB	1024 MiB	513 MiB	1024 MiB
Battery capacity (Ah)	1.4	1.3	1.46	1.5	X	1.5	1.5
Expandable memory	✓	✓	✓	✓	✓	✓	✓
NFC	X	X	X	X	X	X	X
Flash	X	X	X	X	X	X	X
GPS	✓	✓	X	✓	✓	✓	✓
Gyro	X	X	X	X	X	X	X

Table 5, Smart phones

Mobile Phone	LG GW620	LG Optimus 7	LG OPTIMUS BLACK	LG Optimus 1 Google™	Motorola Milestone 2	Motorola Dext	Motorola Atrix
Price (£)	100	349.99	431.99	139.99	367.69	175	499.99 Due
Operating systems	Android 1.5	windows 7	Android 2.2	Android 2.2	Android 2.2	Android 1.5	Android 2.3
Processor	528 MHz Qua lcomm M SM7200 AARM11 processo r	Qualcom m QSD86 50 1GHz Scorpion (Snapdra gon)	1 GHz Nvidia Tegra 2 ARM Cortex-A8	GSM 600 MHz Qua lcommM SM7227 ARM113 6EJ-S	1 GHz OM AP 3620[6]	Qualcom m MSM720 0A	Tegra 2 dual-core 1 G Hz (Cortex-A9 dual-core)
Processor speed (GHz)	0.528	1	1	0.6	1	0.528	1
GPU	X	X	X	Adreno 200	X	X	GeForce ULP
Touch screen	Resistive	Capacitiv e	Capacitiv e	Capacitiv e	Capacitiv e	Capacitiv e	Capacitiv e
Screen size (inch)	3	3.8	4	3.2	3.7	3.1	4
RAM	X	512MB	512MB	512MB	512 MB	256MiB	1GB
Memory	X	16GB	2GB	512MB	8 GB	512 MiB	16GB
Battery capacity (Ah)	X	1.5	1.5	1.5	1.4	1.42	1.93
Expandable memory	X	X	✓	✓	✓	✓	✓
NFC	X	X	X	X	X	X	X
Flash	X	✓	✓	X	✓	X	✓
GPS	✓	✓	✓	✓	✓	✓	✓
Gyro	X	X	✓	X	X	X	X

Table 6, Smart phones continue

Mobile Phone	Motorola Defy	Samsung Omnia 7	Samsung Omnia II	Samsung Galaxy ACE	Samsung Galaxy Gio	Samsung Galaxy S	Samsung I900 Omnia
Price (£)	269.99	509.99	259	229.99	187.99	353.94	493.5
Operating systems	Android 2.1	Windows phone 7	Windows Mobile 6.1	Android 2.2	Android 2.2	Android 2.3.4	Windows Mobile 6.1
Processor	Texas Instruments OMAP 3610	Qualcomm QSD8250 1GHz Scorpion (Snapdragon)	Samsung S3C6410 ARM 1176 800 MHz processor, dedicated graphics	800 MHz Qualcomm MSM7227, ARM11 (Overclocked)	Qualcomm MSM7227	Samsung Hummingbird S5PC110, 1 GHz (Overclock to 1.5GHz)	624MHz Marvell PXA312 processor.
Processor speed (GHz)	0.8	1	0.8	0.8	0.8	1.5	0.624
GPU	PowerVR SGX530	X	X	Adreno 200	X	PowerVR SGX 540	X
Touch screen	Capacitive	Capacitive	Resistive	Capacitive	Capacitive	Capacitive	Resistive
Screen size (inch)	3.7	4	3.7	3.5	3.2	4	3.2
RAM	512 MB	512MB	256MB	384 MiB	X	512 MB	128 MB
Memory	2 GB	8 /16 GB	8 / 16 GB	512 MB	158 MB	8-16 GB	256 MB
Battery capacity (Ah)	1.54	1.5	1.5	1.35	1.35	1.5	1.44
Expandable memory	✓	X	✓	✓	✓	✓	✓
NFC	X	X	X	X	X	X	X
Flash	✓	✓	✓	X	X	X	✓
GPS	✓	✓	✓	✓	✓	✓	✓
Gyro	X	X	X	X	X	✓	X

Table 7, Smart phones continue

Mobile Phone	HTC 7 Pro	HTC Desire HD	HTC Desire S	HTC Desire Z	HTC HD7	HTC Incredible S	HTC Wildfire S
Price (£)	418.25	362.99	349.99	264.99	275.38	365.46	208.99
Operating systems	Windows	Android 2.3	Android 2.3	Android 2.2	Windows 7	Android 2.3	Android 2.3
Processor	Qualcomm QSD8650 1 GHz Scorpion (Snapdragon)	Qualcomm MSM 8255 1 GHz Scorpion (Snapdragon)	1 GHz Qualcomm MSM8255 (Snapdragon)	800 MHz Qualcomm MSM7230Snapdragon	(Qualcomm Snapdragon QSD8250)1 GHz Scorpion	1 GHz Qualcomm SnapdragonMSM8255	Qualcomm MSM7227 600 MHz
Processor speed (GHz)	1	1	1	0.8	1	1	0.6
GPU	X	Adreno 205	Adreno 205	X	Adreno 200	Adreno 205	Adreno 200
Touch screen	Capacitive	Capacitive	Capacitive	Capacitive	Capacitive	Capacitive	Capacitive
Screen size (inch)	3.6	4.3	3.7	3.7	4.3	4	3.2
RAM	576MB	768MB	768MB	512MB	576MB	768MB	512MB
Memory	8GB	1.5GB	1.1GB	1.5GB	16GB	1.1GB	512MB
Battery capacity (Ah)	1.5	1.23	1.45	1.3	1.23	1.45	1.23
Expandable memory	X	✓	✓	✓	X	✓	✓
NFC	X	X	X	X	X	X	X
Flash	✓	✓	✓	✓	✓	✓	✓
GPS	✓	✓	✓	✓	✓	✓	✓
Gyro	X	X	X	X	X	X	X

Table 8, Smart phones continue

Mobile Phone	iPhone 4	Samsung Galaxy S II	Samsung Nexus S	HTC Sensation	LG Optimus 2X	BlackBerry Torch 2
Price (£)	535.99	499.99	322.26	499.99	369.99	X
Operating systems	iOS 4	Android 2.3	Android 2.3	Android 2.3	Android 2.2	RIM BlackBerry 7 OS
Processor	Apple A4	1.2 GHz dual-core ARM Cortex-A9[2] SoC processor; Samsung Exynos (GT-i9100)	1 GHz Samsung HummingbirdS5PC110 (ARM Cortex A8)	1.2 GHz, dual core QualcommMSM 8260 Snapdragon	1 GHz Nvidia Tegra 2 ARM Cortex-A9 dual-core	Dual core
Processor speed (GHz)	X	1.2	1	1.2	1	1.2
GPU	Apple A4 (PowerVR SGX 535)	ARM Mali-400 MP (GT-i9100)/ GeForce ULP (8 cores) (GT-i9103)	PowerVR SGX 540	Adreno 220	ULP GeForce	X
Touch screen	Capacitive	Capacitive	Capacitive	Capacitive	Capacitive	Capacitive
Screen size (inch)	3.5	4.27	4	4.3	4	3.2
RAM	512MB	1GB	512 MB	768MB	512MB	768 MiB
Memory	16GB or 32GB	16 GB	16 GB	1GB	8GB	8 GB
Battery capacity (Ah)	1.42	1.65	1.5	1.52	1.5	1.27
Expandable memory	X	✓	X	✓	✓	X
NFC	X	X	✓	X	X	✓
Flash	✓	✓	✓	✓	✓	✓
GPS	✓	✓	✓	✓	✓	✓
Gyro	✓	✓	✓	✓	✓	X

Table 9, Smart phones continue

Where an X is found on the tables instead of the name of the processor, the battery capacity, the price, the processor speed, the GPU, the memory or the RAM it means

that the information could not be found or it is not available. If an X is found under any other category it means that the device does not have that capability or type of sensor or actuator and if a check (✓) is found, it means that the device does have that capability, sensor or actuator.
The smart phones that have been shown on the tables above have 2 sensors and 3 different communication systems in common. The sensors that are in common are the camera and accelerometer. The three communication systems in common are the Bluetooth, 3G and Wi-Fi.
The prices of the smart phones were taken from Amazon.co.uk. The devices were sold by Amazon as new and SIM free meaning that they were not locked to any mobile operator network.
It can be seen that there is a large price range varying from £100.00 for an LG mobile up to £535.99 for the iPhone 4. It is also noted that the largest price range for OS is the Android OS where a phone can be found from £100.00 up to £499.99, the next largest is Windows OS based devices where a phone can be bought from £259.00 running Windows Mobile OS and up to £509.99 for devices running Windows Phone 7 OS. Windows is followed by BlackBerry OS and iOS, this last operating system has only one phone to offer. The most expensive phones are normally the most recent phones with the latest advances and the best features out in the market. Popularity and brand name influences the final price leading to the most expensive phone not necessarily the best in the current market.
The phones available in the tables above use 4 different operating systems Android, iOS, BlackBerry and Windows. Each operating system has different versions. The most advanced phones are running the latest version of those operating systems. Some of the devices can be upgraded as soon as the new version is available and others have to wait for the software coming with the phone to be updated by the manufacturers before receiving the upgrade.
As can be seen on tables 5 to 9 there are different types of processors, most of the smart phones use single core processor but new release phones use dual core processors. Most of the processors are systems on a chip and the CPU architectures use on those devices is ARM. The processing time varies from phone to phone even

if the same processor has been used in various devices. Also the processing speed depends on the type of architecture that has been used to build the processor.
More than half of the mobile phones shown on the tables above are using Qualcomm processors and within them the Qualcomm, Snapdragon MSM8255 chip is the most used processor. HTC phones shown in the tables all use Qualcomm processors as do Sony devices, 3 Samsung and 3 LG phones use the same manufacturer for the processor but only 1 BlackBerry and 1 Motorola with a Qualcomm processor. Devices using the MSM8255 are the HTC Desire S, the HTC Incredible S, the Sony Xperia arc and the Sony Xperia pro these four phones work at 1 GHz and are using Android 2.3. The Qualcomm Snapdragon MSM8255 has a RISC base CPU with a Qualcomm Scorpion core, the word width is 32 bit and the instruction set supported is the ARMv7. It is a system on a chip with a single Scorpion core CPU running up to 1.4 GHz, an Adreno 205 GPU, it has integrated 3G, it supports HSPA+ networks, Standalone-GPS and Assisted-GPS modes, support for Wi-Fi, Bluetooth and FM connectivity. It also includes the APQ8055 processor and supports Android OS, Brew Mobile Platform, HP webOS2.0 and Windows Phone [17].
The Qualcomm Snapdragon MSM8260 is used in the HTC Sensation. This is a third generation Qualcomm processor. It has a dual Qualcomm Scorpion Harvard Superscalar processor with a processing speed of up to 1.5 GHz. The CPU has a RISC structure using a 32 bit width word with a RAM data bus of also 32 bit and ARMv7 instruction set. The graphics processing unit is the new Adreno 220 with a performance of 4 times the first Adreno 200. The multimedia capabilities have been improved and now it is able to record and reproduce 1080p videos, it supports a 16 megapixel dual camera, 5.1 Dolby surround sound and HDMI mirroring connectivity. The MSM8260 processor has integrated 3G, single mode support for HSPA+, GSM, GPRS and EDGE, Standalone-GPS and Assisted-GPS modes, Wi-Fi, Bluetooth and FM connectivity and includes the APQ8060 processor. It supports Android OS, Brew Mobile Platform, HP webOS2.0 and Windows Phone [17].
Samsung uses its own processors in four of the listed phones. The latest release from Samsung was the Samsung Galaxy SII which has a dual core processor Samsung Exynos running at 1.2 GHz. The Exynos processor is a system on a chip, the architecture of the device can be seen on figure 5 below. It has a dual core Cortex A9

with 1MB L2 Cache and the Mali 400MP GPU for the GT I9100 model or the GeForce ULP 8 cores GPU for the GT I9103 model. Cortex A9 it is claimed to be 25% DMIPS faster than its predecessor Cortex A8. With its 6.4GB/s memory bandwidth it can handle operations such as 1080p video encoding and decoding, 3D graphics display, and it can also handle a triple display, supporting WSVGA LCD dual display and 1080p HDMI, simultaneously [18].

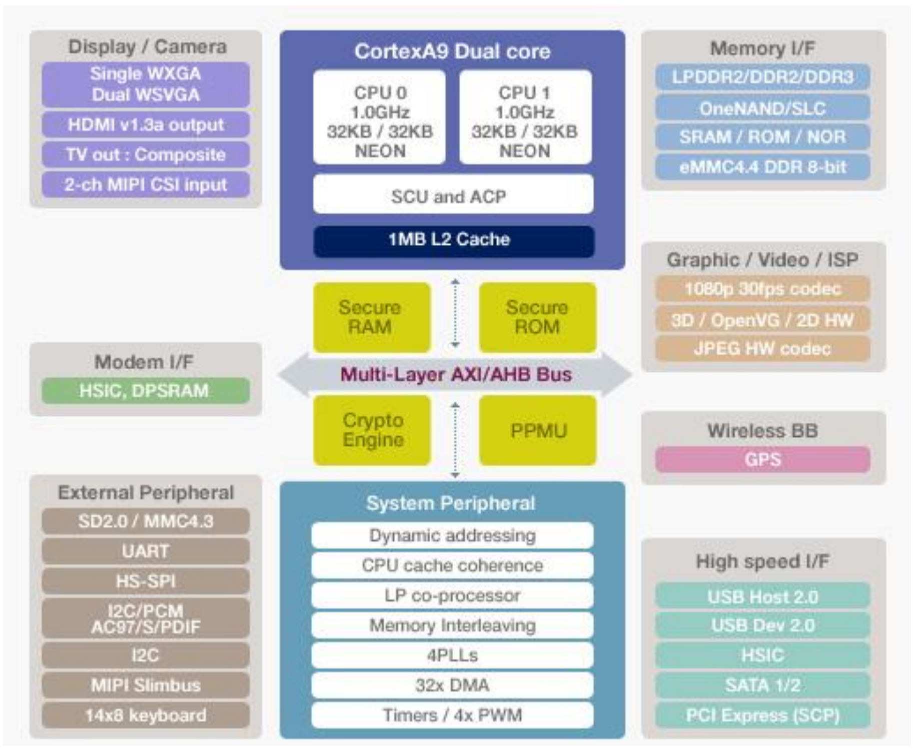


Figure 5, Exynos processor [18]

The Samsung Hummingbird S5PC110 is used in two different Samsung phones running at different speeds, the Nexus S working at 1 GHz and the Galaxy S working at 1.5 GHz. The Hummingbird S5PC110 uses a single core ARM Cortex A8 with PowerVR SGX540 GPU, 512KB L2 Cache and 32KB instruction plus 32KB of data in the L1 cache. It also uses ARM v7 instruction set. It has 96KB of internal RAM and 64KB of internal ROM. When first released in 2009 it had the fastest processing speed on the market. Some of its features are a 64-bit multi-layer bus architecture, advanced power management for mobile applications, multi format CODEC provides encoding and decoding of up to 1080p at 30fps, a 3D Graphics Accelerator, dedicated 2D Graphics accelerator, and the memory subsystem supports asynchronous memory interface, LPDDR1 interface, DDR2 interface and LPDDR2 amongst other features.

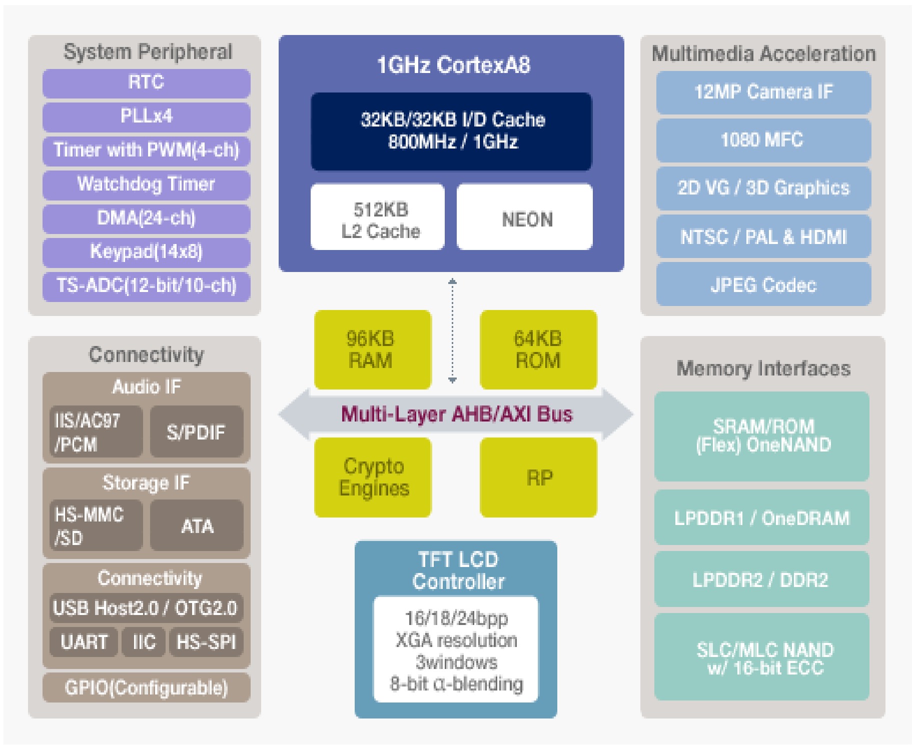


Figure 6, S5PC110 processor [19]

LG Optimus 2x and the Motorola atrix use the same processor, the Nvidia Tegra 2 which is a dual core processor operating at 1.2 GHz with two ARM Cortex A9 CPUs, 1MB L2 cache and 32KB L1cache per core. Uses an ultra low power GeForce GPU with 8 cores, openVG 1.1 and EGL 1.4 and it can decode and encode video up to 1080p. It supports 2 displays simultaneously through HDMI, LCD and CRT [20].

The iPhone 4 uses the Apple A4 processor. It uses a single core ARM Cortex A8 processor, the system on a chip was designed by Apple and manufactured by Samsung. It uses a similar GPU to the Hummingbird processor, the PowerVR SGX535. The processing speed of the iPhone 4 has not been released but the processor could go up to 1GHz or higher if over clocked. As for the Hummingbird the Apples A4 uses ARM v7 instruction set, 512KB L2 Cache and 32KB instruction plus 32KB of data for the L1 cache.

There are processors from different companies like Texas Instruments and Marvel which are integrated into devices like the Motorola Defy, Motorola Milestone 2, the Samsung I900 Omnia and the BlackBerry torch.

There are two technologies used for touch screens resistive technology and capacitive technology. The differences between these two technologies are:

- Resistive: resistive touch screens can be used with the finger, a stylus or any other object. In general are less responsive and accurate when using bare fingers than when using a stylus. Resistive touch screens are passive devices that use the principle of the voltage divider to calculate the touch point on the screen[21].
- Capacitive: capacitive touch screen are active devices and only work with a conductive pointer such as a finger. There are different technologies for these types of devices, surface capacitance and projected capacitance. Projected capacitance is split into two, mutual capacitance and self-capacitance. Mutual capacitance is the most accurate of the technologies. Capacitive touch screens are more sensitive and accurate than resistive touch screens [21].

Of the 34 smart phones reviewed there are only three that use the resistive technology, the rest use capacitive technology. Few applications in health take advantage of the touch screen capability as an input tool for measuring performance. The size of the screens varies between 3 inches diagonally up to 4.3 inches for the larger devices.

The amount of RAM memory that can be found within the devices goes between 128 MB and 1GB. There are two phones (the Samsung I900 Omnia and BlackBerry Storm) that have 128MB , the Motorola Atrix and the Samsung Galaxy S II have 1GB of RAM. It can be notice that the dual core processors are using the 1GB RAM memories.

The internal memory of the devices varies between 158MB up to 32GB. Devices that have less than 8GB of internal memory have the possibility of expanding the memory with an external card, depending on the device the external card that can be used could be up to 32GB. There are 13 devices with 8GB or more of internal memory. 6 devices out of those 13 have the possibility of further expanding the memory but 7 of those 13 phones do not have the possibility of expanding it. Those devices can be seen on the table below.

Mobile Phone	Memory	Expandable memory
Samsung Omnia II	8 / 16 GB	✓
Motorola Milestone 2	8 GB	✓
LG Optimus 2X	8GB	✓
Motorola Atrix	16GB	✓
Samsung Galaxy S	8-16 GB	✓
Samsung Galaxy S II	16 GB	✓
BackBerry Torch 2	8 GB	X
iPhone	16GB or 32GB	X
HTC 7 Pro	8GB	X
LG Optimus 7	16GB	X
Samsung Nexus S	16 GB	X
Samsung Omnia 7	8 /16 GB	X
HTC HD7	16GB	X

Table 10, Greater than 8 GB internal memories

The capacity of the battery is measured in Ah in the tables 5 to 9. The devices with the least capacity are the HTC Wildfire S, HTC Desire HD and the HTC HD7 with a capacity of 1.23 Ah. The Motorola Atrix has the largest battery capacity, 1.93Ah. The battery capacity affects the autonomy of the device; a device with bigger capacity should have more autonomy than a device with smaller capacity. The autonomy is also affected by the type and speed of the processor and the power management of the device.

There are only two mobile devices that have near field communication (NFC) the BlackBerry Torch and Samsung Nexus S. NFC can be used to perform credit card payments, be used as an ID card, an electronic key, to read information from a tag, it can share data with other NFC devices or even pair a Bluetooth or wireless connection. For these communications to happen, devices must be within a distance of about 4cm or less and the introduction of a PIN for some of the uses maybe required [22]. The use of NFC linkage to medical monitors may be of some practical use and could offer some security benefits related to patient confidentiality.

Seven of the shown smart phones use a gyroscope. Gyroscopes are used to measure orientation; together with the accelerometer it can tell the direction and speed of the device.

Some of the devices come with a flashlight. The flashlight of the smart phones is normally made from LEDs.

Almost all of the smart phones are fitted with a GPS antenna with the exception of the Casio G'zOne Commando. The GPS can give the position of the device within a few meters of error.

The Casio G'zOne Commando it designed to be a tough phone that can survive being dropped, immersed in water and is specified to work in extremes of temperature and harsh environment.

3. Platform recommendation

This section aims and objectives were:

- To compare the medical applications found within the different OS
- To compare the various operating systems
- To compare the characteristics of the smart phones
- To conclude with a recommendation of a platform and a smart phone
- To propose a medical application

3.1. Medical applications comparison

As shown on tables 1, 2 and 3 the medical related applications found in the different stores and market places have different prices. There are 12 applications that are available for free within the Android market and 11 free applications can be found in the iOS store. The total amount of applications found was 40 in the Apple store, 18 in the Android market and 2 in the BlackBerry app world.

Within the 12 free applications found within the Android market, 9 are used to monitor the heart. Only 2 out of the 9 applications use a sensor within the phone, the other 7 applications use an external device to measure the heart rate. The sensor used to measure the heart rate is the camera which detects changes in the colour of the skin. The measurements of the applications are not very accurate as can be understood from the reviews left by the users in the android market.

The heart monitoring applications that use external devices connect to the phones using Bluetooth communications. The average user rating for most of these applications is very low although there are not many user ratings in most of them. The Cardio Trainer app has the highest average rating, given by the high number of users. This application uses the GPS and accelerometer sensors from the phone and connects to the internet. It claims that it is the first application that has partnered with Google Health. However, its future given the demise of Google Health is unknown.

Response Measurement is one of the free applications developed for Android OS. It is designed to measure the response time to a stimulus, it was developed for patients with Parkinson’s disease or movement disorders. The application works as follows, when the patient or user is ready they have to press the start button, then four squares out of nine are lit with different colours. The user has to press the coloured squares one by one as fast as possible. This has to be done a number of times to generate

enough information to make it possible to create three different charts or graphs. The results indicated in the graphs are then studied by a clinician that decide if the person has Parkinson’s disease, motor or cognitive disorder or if the treatment is the adequate. There are only two reviewers and have given to the app the maximum rating.
There are two applications listed dedicated to blood glucose, both of them use external devices to measure the sugar levels and transfer this information to the phone via Bluetooth. Glucose Meter has been given a high rating by numerous users but Bluetooth Blood Glucose has not received any ratings.
Other applications found aim to relieve the symptoms of tinnitus, another is a hearing test, Smart Hearing Aid which amplifies the sound helping the user to hear more clearly and the last application which uses an external scale to control the weight of the user.
There are only 4 applications out of the 11 free apps produced for the iPhone that use external sensors to gather the information, the other 7 apps use the sensors available in the phone.
Bam Labs, RVG Mobile, Mobile MIM and WiScale are the four applications that use external devices. Bam Labs is used to monitor the users sleep. RVG Mobile is used for image acquisition; it works with the Kodak RVG 6500 System that is used to take x-rays. Mobile MIM is designed to reproduce images taken by different x-ray devices. And WiScale that uses a scale to weigh the user and monitor blood pressure.
There are 3 of the 11 free applications that use the camera to monitor the heart rate. Other 3 applications use the speaker phone or the headphones to perform a hearing test and the Motor application that uses the accelerometer to check the movements of the patient.
Only 1 out of the pay applications found in the iOS market used an external device to gather data. SporTraker is the name of the app and it uses Bluetooth for the communications between the phone and the external device, it also uses the built-in GPS and accelerometer.
The rest of the applications only use the sensors that are available in the phone.
There are applications that can be found within the Android market and the App Store.

It can be noticed that there were more than double the number of medical related applications in the App Store than in the Android market. Most of the Android applications were free but required an external device to work. A charge was made for the majority of the applications found in the App store but no external device was required.

The most popular applications found within both markets were the ones checking the heart rate, blood glucose, hearing testing, hearing aids, tinnitus and weight and sports related applications.

There were 3 applications found that relate to motor functions to follow treatment or to help with the diagnosis of motor function diseases or problems like Parkinson’s disease. Those applications are Response Measurement developed for Android, Tremor Tracer and Motor that are developed for the iPhone. Two applications, the Response Measurement and the Tremor Tracer use the screen to gather the information required to help with the diagnosis or to follow the treatment. Tremor Tracer uses a common technique use by the clinicians to test patients for Parkinson’s disease. The patient has to trace a spiral and a straight line on the screen of the device. The Tremor Tracer application has two reviews both showing it to be weak. The Response Measurement application was explained before, this application measures the time that the user takes to touch each of the lit squares.

As shown there are not many medical applications that use the touch screen of the devices as a sensor to gather the data to help in the diagnosis and or follow the treatments. The information that can be retrieved from the screen could come from multiple points touch in the screen and the information that could be gathered from the multiple pointers is the time at which each pointer has touch the screen, the number of pixels that have been pressed in each of the taps, the number of pointers that are touching, the time at which each pointer has been retrieved, which pointer has been retrieved, the exact location at which the pointer has touched the screen and if that pointer only touches the screen or if it has movement. At the moment there are no applications which use all these features which could indicate how repetitive, accurate, fast and strong a tap or multiple taps are on the screen. Those measurements could have been used to develop applications for the management of many forms of motor and cognitive impairment.

3.2. Operating Systems Comparison

There is only one OS out of the four reviewed, the Android OS, which has all the software needed to program applications available to be downloaded for free without requiring a permit.

The base language to develop applications in the different languages is C and C++. Android uses Java with new commands just for Android, iOS uses Objective C, BlackBerry uses C++ or Java and Windows Mobile 7 uses Metro.

All the platforms give simulation tools where to test the application before deploying it onto the phone. Android OS lets the testing of applications on any device at any time, iOs gives permission to test an application on a device as long as the programmer has a license. To publish an application on any of the application markets requires a registration fee and then there is a percentage charge per each sale unless the applications are free of charge to the public.

Android OS gives all the required information, tutorials and sample code for free in their web site as do iOS for the development of their applications.

The worldwide market share comparison shown on figure 4 states that the operating system that is going to grow the most in the next year or so is the Android OS follow by Microsoft. The market shares of BlackBerry OS and iOS are going to fall. By 2015 Gatner is expecting the market share to be 48.8% for Android OS, 19.9% for Microsoft and for iOS to have fallen to 17.2% and RIM to 11.1%. Figures 2 and 3 also show that the Android market share in each of the six countries has grown at least 11 times of that in the previous year.

This comparison shows that the Android OS is the most popular operating system in the market, the easiest and cheaper to get access and it can be distributed in an independent market. The Android OS is being developed and changes occur at all times making it difficult to find any books and or tutorials from third party authors where to learn to program applications.

3.3. Smart phones comparison

There were 34 smart phones compared in tables 5 to 9, only 6 of them were considered for this final review. These phones were the iPhone 4, Samsung Nexus S, BlackBerry Torch 2 (not jet released), LG Optimus 2x, HTC Sensation and Samsung Galaxy S II. These six phones were chosen for their characteristics. They have features in common, a capacitive touch screen, a GPS, Flash light source, a camera, accelerometer, Bluetooth, 3G and Wi-Fi. The operating systems that these phones use are the iOS 4, the Android 2.3, Android 2.2 and RIM BlackBeery 7 OS. The BlackBerry Torch2, LG Optimus 2x, HTC Sensation and Samsung Galaxy S II use dual core processors running at over 1GHz processing speed. The phone with the largest RAM memory is the Samsung SII with 1GB. The largest screen is 4.3 inches across, which is the HTC Sensation. The latest generation GPUs are being used in the dual core phones therefore the phones can process faster and more complex images.

The Samsung Galaxy SII internal memory has 16GB of capacity which can be expanded with an external memory card, a gyroscope and the battery capacity is the 1650 mAh. The operating system is the Android 2.3.

The HTC Sensation has a 1GB internal memory which can be expanded with an SD card, 768MB of RAM, a gyroscope and its battery capacity is the 1520 mAh. The operating system is the Android 2.3.

The LG Optimus 2x internal memory is the 8GB expandable with an external memory card, 512 MB of RAM, gyroscope and 1500 mAh battery capacity. The operating system is the Android 2.2.

The iPhone 4 and Samsung Nexus S have single core processors with a Power VR SGX GPU, the Nexus S uses the Power VR SGX 540 GPU and the iPhone the Power VR SGX 535 GPU. The processing speed of the iPhone 4 has not been released but the Samsung Nexus S processing speed is 1GHz. Both phones have the same amount of memory RAM 512MB. The internal memory of the iPhone can be 16 or 32 GB and it cannot be expanded. The Nexus S internal memory is 16GB and as with the iPhone it cannot be expanded. The Nexus S has a larger screen size of 4 inches and larger battery capacity of 1500mAh compared to the iPhone 4 which has 3.5 inches screen and 1420mAh battery capacity. Both phones have a gyroscope but only the Samsung Nexus S has near field communications.

The Samsung Nexus S is pure Google mobile which runs only Android, this means that any updates made to the OS can be downloaded into the phone as soon as they are available.

The iPhone 4 uses iOS 4 which can also be updated as soon as Apple releases a new update compatible with the iPhone 4.

These 6 devices are at the high end of the market as well as their prices. The prices found for the phones in the Amazon web site were from highest to lowest- iPhone £535.99, Samsung Galaxy S II £499.99, HTC Sensation £499.99, LG Optimus 2x 369.99, Samsung Nexus S £322.26 and the BlackBerry Torch 2 did not have a price yet.

As shown the best overall specified phone out of these 34 smart phones is the Samsung Galaxy S II as it has a dual core processing at 1.2GHz, an ARM Mali 400MP GPU, a 4.27 inch screen, 1GB of RAM and a 16GB of internal memory which can be expanded. It also is one of the phones receiving the best reviews for smart phones [23].

The most economical of the chosen smart phones is the Samsung Nexus S which it comes equipped with everything except the dual core processor. It is a good phone for developers as it is cheaper than the iPhone, it can have the latest version of Android OS installed, does not required an external memory and has all the sensors and communications systems available in the market. It has been place in the position number 8 in the ZDNet [23].

3.4. Smart phone recommendation conclusion

The recommended OS platform was the Android OS as it is the most popular and the most easily accessible operating system in the market. It was not necessary to invest in any other devices apart from the smart phone to start the application development.
The recommended smart phone was the Samsung Galaxy S II which comes with Android 2.3 and as explained in section 3.3., is the best specified phone available at the moment in the market.
Due to the price of the Samsung Galaxy S II a second recommendation was given. The Samsung Nexus S was recommended for this project. This phone has all the communications systems available in smart phones at the moment and all the sensors and actuators available in smart phones in the current market.

3.5. Medical application scoping

As explained earlier there are only two medical applications that use the touch screen capabilities of the devices as an input method. In movement disorders the measurement of repetitive movements such as alternating finger tapping can provide clinically useful data on a patient's disease symptoms and disease progression. An example of such a test is Quantitative Digitography (QDG). QDG is a test developed and evaluated by Dr Helen Bronte-Stewart and her team at Stanford University. It is used to help in the diagnosis and to follow the treatments of patients with Parkinson's disease. QDG was developed using a MIDI keyboard attached to a computer to record different parameters during repetitive, alternating finger tapping sequencing. QDG is often used by the Stanford group in measuring changes in Parkinson's disease patient status prior to and after Deep Brain Stimulation surgery [24].

The parameters measured with this technique are as follows, the duration time of each key strike, the key that was struck, the velocity of each strike, the interval between strikes and the time of each strike [24]. In relation to smart phone technology these measurements could be gathered by developing an application that uses the touch screen as a sensor. As explained earlier, the information that can be obtained from alternate tapping with two fingers on to the screen is similar to the information that can be obtained from striking a MIDI keyboard. The time at which each finger touches the screen and releases the screen can be obtained giving the duration of each strike and the time interval between each strike can also be measured. The velocity of each strike can relate to the number of pixels that have been pressed during each strike. In addition, information could be added like the exact position on the screen of each strike to check for repetitiveness and accuracy. It is also possible to check for a movement of the finger while in contact with the screen.

QDG therefore would appear to be a good candidate test that may be able to be implemented as an application on smart phone platforms.

It has been shown to be a sensitive measure of motor control in patients [25] and results correlate with the most common Parkinson's Disease rating scale (UPDRS) Importantly, in patients undergoing deep brain stimulation it has been used to quantify improved motor function in response to the treatment [24]. As a smart phone application, simulation of the QDG test would make use of the inherent ability of the majority of smart phones good graphics and touch screen interactivity.

4. Medical Device Directive

The aims and objectives of this section were:

- Categorise the smart phone and the application within the classes proposed by the Medical Device Directive
- To establish the requirements needed to get a CE mark

4.1. Regulations

The application and the smart phone falls under the definition stated in the Medical Device Directive (MHRA) for medical devices. The definition from the MHRA states as the following:

> *“'any instrument, apparatus, appliance, material or other article, whether used alone or in combination, including the software necessary for its proper application intended by the manufacturer to be used for human beings for the purpose of:*
>
> - *diagnosis, prevention, monitoring, treatment or alleviation of disease,*
> - *diagnosis, monitoring, treatment, alleviation of or compensation for an injury or handicap,*
> - *investigation, replacement or modification of the anatomy or of a physiological process,*
> - *control of conception,*
>
> *and which does not achieve its principle intended action in or on the human body by pharmacological, immunological or metabolic means, but may be assisted in its function by such means.'”[26]*

There are four levels of risk in which the application and the smart phone can be classified. In this case the medical device application and the device are low risk and can be classified into Class I of the Medical Device Directive. This is been determined by rule 1 and rule 12. Rule 1 applies to all non-invasive devices and places them into Class I and rule 12 applies to all active devices that cannot be categorized under rule 9, rule 10 and rule 11 and places them into Class I.

As explained in the Medicine and Healthcare products Regulatory Agency (MHRA) website, the essential requirements of the Medical Device Directive can be certified by the registered manufacturer alone, only if the device has been classified into Class I. But if the Class I device is sterile or has a measurement function a Notified Body must be involved to give the CE marking [27].

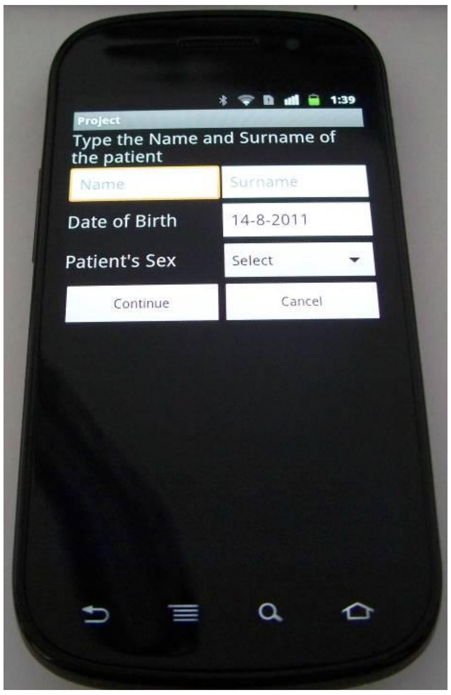
5. Software development for Smart phone implementation of Quantitative Digitography.

This section aims and objectives were:

- To design the windows needed in the application
- To write the code for the application
- To test that the software created was working as expected

The application programming language was Java as recommended for Android OS and the target phone was the Samsung Nexus S. The programs required for the programming of the application were the Java Development Kit (JDK), Eclipse IDE, Android SDK and Eclipse ADT plug-in.

Three windows were created. When the application is first started the first window that appears asks for the name, surname, date of birth and sex of the person to be tested to be inserted as shown in the figure 7 below.



Type the Name and Surname of the patient

Date of Birth 14-8-2011

Patient's Sex

Figure 7, First page

To create this window the following code was created.

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView
xmlns:android="http://schemas.android.com/apk/res/android"
android:orientation="vertical"
android:layout_width="match_parent"
```

```
android:layout_height="match_parent">

<LinearLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
                android:orientation="vertical"
                android:layout_width="match_parent"
                android:layout_height="match_parent">
  <TextView
  android:text="@string/PatientsName"
  android:id="@+id/PatientsName"
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  android:textSize="20dip"
  </TextView>

        <LinearLayout
                xmlns:android="http://schemas.android.com/apk/res/android"
                android:orientation="horizontal"
                android:layout_width="match_parent"
                android:layout_height="match_parent">

                <EditText
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"
                        android:id="@+id/PatientsNameText"
                        android:hint="@string/PatientsNameText"
                        android:layout_weight="1"
                        android:inputType="textPersonName"
                        >
        <requestFocus>
        </requestFocus>
                </EditText>

        <EditText
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"
                        android:id="@+id/PatientsSurnameText"
                        android:hint="@string/PatientsSurnameText"
                        android:layout_weight="1"
                android:inputType="textPersonName"
                >
                </EditText>
        </LinearLayout>

        <LinearLayout
                xmlns:android="http://schemas.android.com/apk/res/android"
                android:orientation="horizontal"
                android:layout_width="match_parent"
                android:layout_height="match_parent">

                <TextView
                        android:text="@string/DateOfBirth"
                        android:id="@+id/DateOfBirthT"
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"
                        android:textSize="20dip"
                        android:layout_weight="1"
                ></TextView>

                <EditText
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"
                        android:id="@+id/DateOfBirthET"
                        android:hint="@string/DateOfBirthH"
                        android:layout_weight="1"
                        android:inputType="date"
                ></EditText>
</LinearLayout>

        <LinearLayout
                xmlns:android="http://schemas.android.com/apk/res/android"
                android:orientation="horizontal"
                android:layout_width="match_parent"
                android:layout_height="match_parent">

                <TextView
                        android:text="@string/PatientsSex"
                        android:id="@+id/PatientsSex"
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"
                        android:textSize="20dip"
                        android:layout_weight="1"
                >
                </TextView>

                <Spinner
                        android:id="@+id/PatientsSexSpinner"
                        android:layout_width="match_parent"
                                android:layout_height="wrap_content"
                        android:prompt="@string/PatientsSexSpinner"
                        android:layout_weight="1"
                >
                </Spinner>

        </LinearLayout>

        <LinearLayout
                xmlns:android="http://schemas.android.com/apk/res/android"
                android:orientation="horizontal"
                android:layout_width="match_parent"
                android:layout_height="match_parent">
                <Button
                        android:text="@string/ContinueButton"
                        android:id="@+id/ContinueButton"
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"
                        android:layout_weight="1"
                        android:clickable="true">
                </Button>

                <Button
                        android:text="@string/CancelButton"
                        android:id="@+id/CancelButton"
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"
                        android:layout_weight="1"
                >
```

```
                    </Button>
  </LinearLayout>
</LinearLayout>
</ScrollView>
```

This part of the code it is the .xml file, used to show how the window looks but it does not implement any function. The behaviour of the views that are in the window are written in a .java file shown below.

```
package com.Strath.Project;
import java.util.Calendar;
import android.app.Activity;
import android.app.DatePickerDialog;
import android.app.Dialog;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.DatePicker;
import android.widget.EditText;
import android.widget.Spinner;
public class ProjectActivity extends Activity
{
        private EditText etName;
        private EditText etSurname;
        private Spinner spinnerS;
        private EditText etDOB;
  private int mYear;
  private int mMonth;
  private int mDay;
  static final int DATE_DIALOG_ID = 0;

  /** Called when the activity is first created. */
  @Override
  public void onCreate(Bundle savedInstanceState)
  {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.firstpage);
```

```
//Set spinner and adapter

spinnerS = (Spinner) findViewById(R.id.PatientsSexSpinner);
final ArrayAdapter<CharSequence> adapterS = ArrayAdapter.createFromResource
(this, R.array.PatientsSexSpinnerArray, android.R.layout.simple_spinner_item);
adapterS.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
spinnerS.setAdapter(adapterS);

//Set Date of Birth
etDOB = (EditText)findViewById(R.id.DateOfBirthET);
etDOB.setOnClickListener(new View.OnClickListener()
{
                        public void onClick(View v)
                        {
                                showDialog(DATE_DIALOG_ID);

                        }
            });
final Calendar Today = Calendar.getInstance();
mYear = Today.get(Calendar.YEAR);
mMonth = Today.get(Calendar.MONTH);
mDay = Today.get(Calendar.DAY_OF_MONTH);
updateDisplay();



Button Continue = (Button)findViewById(R.id.ContinueButton);
Button Cancel = (Button)findViewById(R.id.CancelButton);
Continue.setOnClickListener(new View.OnClickListener()
{
    public void onClick(View view)
    {
            //Get name, surname and date of birth
            etName = (EditText)findViewById(R.id.PatientsNameText);
            etSurname = (EditText)findViewById(R.id.PatientsSurnameText);
            String Name = etName.getText().toString();
            String Surname = etSurname.getText().toString();
            String DOB = etDOB.getText().toString();

            String PosO = (String)spinnerS.getSelectedItem();
```

```
                    if
(Name.equals("")||Surname.equals("")||PosO.equals("Select")||DOB.equals(Today))
                    {
                    Intent popupfirstpage1 = new Intent(ProjectActivity.this, popupfirstpage1.class);
                            startActivity(popupfirstpage1);
                            finish();
                    }
                    else
                    {
                            Intent Activity1 = new Intent(ProjectActivity.this, ProjectActivity1.class);
                            startActivity(Activity1);
                            finish();
                    }
                            }
                    });

      Cancel.setOnClickListener(new View.OnClickListener()
      {

                            public void onClick(View v)
                            {
                                    finish();

                            }
                    });
   }

//updates the date in the TextView
private void updateDisplay()
{
        etDOB.setText
        (new StringBuilder()
        // Month is 0 based so add 1
        .append(mDay).append("-")
        .append(mMonth + 1).append("-")
        .append(mYear).append(" "));
        }
//the call back received when the user "sets" the date in the dialog
```

```
private DatePickerDialog.OnDateSetListener mDateSetListener =
        new DatePickerDialog.OnDateSetListener()
{

        public void onDateSet(DatePicker view, int year, int monthOfYear,
                        int dayOfMonth)
        {
                mYear = year;
                mMonth = monthOfYear;
                mDay = dayOfMonth;
                updateDisplay();

        }
};
@Override
protected Dialog onCreateDialog(int id)
{
        switch (id)
        {
        case DATE_DIALOG_ID:
                return new DatePickerDialog(this,
                                mDateSetListener,
                                mYear, mMonth, mDay);
        }
        return null;
}
}
```

This Java code works as follows, if a name is set, a surname is set and the sex of the patient is set from the drop down menu. On completion the Continue button is pressed the second window appears on the screen. If the Cancel button is pressed at any time the application is closed. If the Continue button is pressed but one or more of the fields have not been introduced or changed the application shows a window asking to complete those fields as shown in figure 10.

Figure 8 shows the menu that appears when the date of birth field is pressed twice and figure 9 shows the drop down menu to choose the sex of the patient.

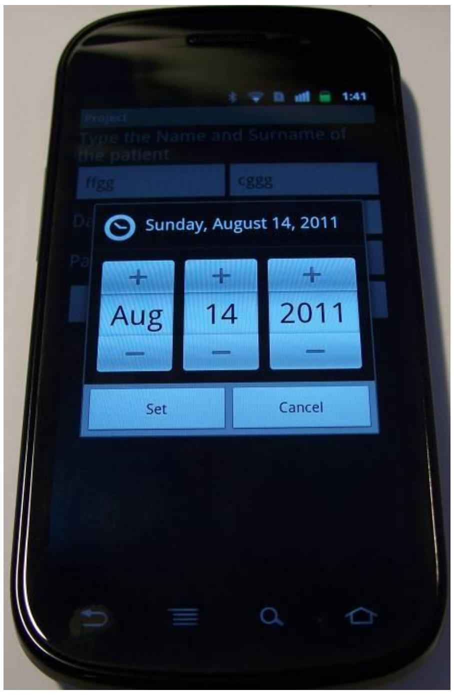


Figure 8, First page date of birth setting

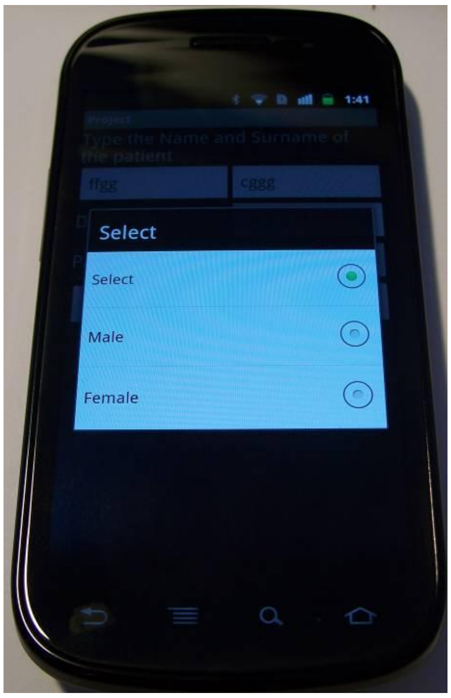
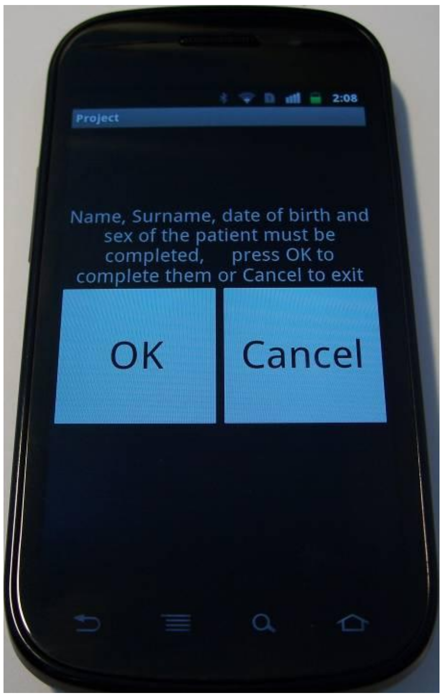


Figure 9, First page patient sex selection



Name, Surname, date of birth and sex of the patient must be completed, press OK to complete them or Cancel to exit

Figure 10, Error page

Each page has its own .xml file and its own .java file. The .xml code for the window shown in figure 10 is very simple as is the .java code. As can be seen from its .java code below when the OK button is pressed the window closes and window on figure 7 appears. If the Cancel button is pressed the application is terminated. In practice, it would not be possible for a patient with severe Parkinson's to enter their own name or personal details, these would have to be done by a clinician or carer.

.xml code:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
 xmlns:android="http://schemas.android.com/apk/res/android"
 android:orientation="vertical"
 android:layout_width="match_parent"
 android:layout_height="match_parent"
 android:background="#00000000"
 >

  <TextView
  android:text="Name, Surname, date of birth and sex of the patient must be completed,
  press OK to complete them or Cancel to exit"
  android:id="@+id/popup1"
  android:layout_width="wrap_content"
```

```xml
    android:layout_height="wrap_content"
    android:textSize="20dip"
    android:gravity="center_horizontal"
    android:layout_gravity="center_vertical|center_horizontal"
            android:layout_marginTop="100dip"
    >
    </TextView>

    <LinearLayout
                xmlns:android="http://schemas.android.com/apk/res/android"
                android:orientation="horizontal"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_gravity="center_vertical">

            <Button
                android:text="OK"
                android:id="@+id/OK"
                android:layout_width="match_parent"
                android:layout_height="150dip"
                android:layout_weight="1"
                android:textSize="40dip"
                android:layout_gravity="center_vertical"
            ></Button>

            <Button
                android:text="Cancel"
                android:id="@+id/Cancel"
                android:layout_width="match_parent"
                android:layout_height="150dip"
                android:layout_weight="1"
                android:textSize="40dip"
                android:layout_gravity="center_vertical"
            ></Button>

    </LinearLayout>
</LinearLayout>
```

.java code:

```
package com.Strath.Project;
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
public class popupfirstpage1 extends Activity
{
            private Button OK;
            private Button Cancel;
             /** Called when the activity is first created. */
   @Override
   public void onCreate(Bundle savedInstanceState)
   {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.popupfirstpage1);

      OK = (Button)findViewById(R.id.OK);
      Cancel = (Button) findViewById(R.id.Cancel);
      OK.setOnClickListener(new View.OnClickListener()
      {
          public void onClick(View view)
          {
                    Intent Activity = new Intent(popupfirstpage1.this, ProjectActivity.class);
                    startActivity(Activity);
                    finish();
          }
      });

      Cancel.setOnClickListener(new View.OnClickListener()
      {
          public void onClick(View view)
          {
                    finish();
          }
      });
   }
}
```

The window in figure 11 appears after pressing the Continue button in the first window. This shows two areas that are the target areas for alternating finger tapping. At the top of the screen two text views are updated each time that one of the buttons are pressed and it reports the number of times that the left and the right buttons have been hit. On the right top corner is a text box with a number representing the time in seconds that the application would run when working. Android does not support multi-touch capabilities between the different views within a window. In this window each button is a different view thus if one of the buttons is pressed the other is disabled and cannot be pressed therefore the corresponding text view does not get updated. The .xml file and the .java file are shown below figure 11.

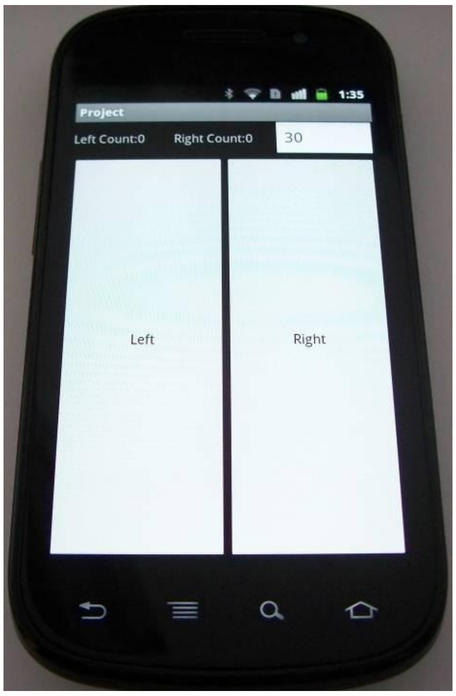


Figure 11, Second page

.xml code:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
            xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_height="fill_parent"
   android:layout_width="fill_parent">

   <LinearLayout
                        xmlns:android="http://schemas.android.com/apk/res/android"
                        android:orientation="horizontal"
```

```
        android:layout_height="fill_parent"
        android:layout_width="fill_parent"
        android:id="@+id/MainLay">

        <TextView
                android:layout_width="match_parent"
                android:textAppearance="?android:attr/textAppearanceSmall"
                android:text="@string/LeftCount"
                        android:layout_weight="1"
                android:id="@+id/LeftCount"
                android:layout_height="wrap_content"
                android:layout_alignParentLeft="true"
                android:layout_marginTop="0dip"
        ></TextView>

        <TextView
                android:layout_width="match_parent"
                android:textAppearance="?android:attr/textAppearanceSmall"
                android:text="@string/RightCount"
                android:id="@+id/RightCount"
                android:layout_height="wrap_content"
                android:layout_marginTop="0dip"
                android:layout_weight="1"
        ></TextView>
        <EditText
                android:id="@+id/TimeDown"
                android:layout_width="fill_parent"
                android:layout_height="wrap_content"
                android:layout_alignParentRight="true"
                android:layout_weight="1"
                android:editable="true"
                android:text="30"
                android:clickable="true"
                android:enabled="true"
                android:numeric="integer"
                        android:cursorVisible="false"
        ></EditText>

</LinearLayout>
```

```
                <LinearLayout
                        xmlns:android="http://schemas.android.com/apk/res/android"
                        android:orientation="horizontal"
                android:layout_height="fill_parent"
                android:layout_width="fill_parent">

                <Button
                        android:text="@string/LeftButton"
                        android:id="@+id/LeftButton"
                        android:layout_width="fill_parent"
                        android:layout_height="fill_parent"
                                android:layout_alignParentLeft="true"
                                android:layout_marginTop="50dip"
                        android:layout_weight="1"
                        android:focusableInTouchMode="false"
                        android:drawingCacheQuality="auto"
                        android:hapticFeedbackEnabled="true"
                        android:soundEffectsEnabled="true"
                        android:capitalize="none"
                        android:visibility="visible"
                ></Button>

                <Button
                        android:id="@+id/RightButton"
                        android:text="@string/RightButton"
                        android:layout_width="fill_parent"
                        android:layout_height="fill_parent"
                                android:layout_alignParentRight="true"
                                android:layout_marginTop="50dip"
                                android:layout_weight="1"
                ></Button>
        </LinearLayout>
</RelativeLayout>
```

.java code:

```
package com.Strath.Project;
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

public class ProjectActivity1 extends Activity
{
        private TextView LeftCount;
        private TextView RightCount;
        private Button LeftButton;
        private Button RightButton;
        private int LN;
        private int RN;
        private EditText SetTime;
        int i;

        @Override
        public void onCreate(Bundle savedInstanceState)
  {

      super.onCreate(savedInstanceState);
      setContentView(R.layout.firstpotrait);

      //Find text views, buttons and edit text

        LeftCount= (TextView) findViewById(R.id.LeftCount);
                RightCount= (TextView) findViewById(R.id.RightCount);
        LeftButton = (Button)findViewById(R.id.LeftButton);
        RightButton = (Button)findViewById(R.id.RightButton);
        SetTime = (EditText)findViewById(R.id.TimeDown);

        clickButtons();
  }
        /**************************************************/
        // programs
        private void clickButtons()
        {
                        LeftButton.setOnClickListener(new View.OnClickListener()
                {

                        public void onClick(View v)
                        {
                                LN = LN+1;
                                updateLeft();
                        }


                });
                RightButton.setOnClickListener(new View.OnClickListener()
                {
                        public void onClick(View v)
                        {
                                RN = RN+1;
                                updateRight();
                        }
                });
        }
        /************************************************/
    private void updateLeft()
    {
        LeftCount.setText(new StringBuilder()
        .append("Left Count:  ").append(LN));
    }
    private void updateRight()
    {
        RightCount.setText(new StringBuilder()
        .append("  Right Count: ").append(RN));
    }
}
```

Figure 12 shows a different second screen for the application, this screen is an unfinished attempt to fix the problem encounter in figure 11. Each time that one of the buttons is tapped a text view appears at the top of the screen indicating which buttons have been pressed, released or moved and if there are one, two or three pointers. If there is only one pointer it gives its exact horizontal position in pixels. The .xml code is show under figure 12 followed by the .java code.

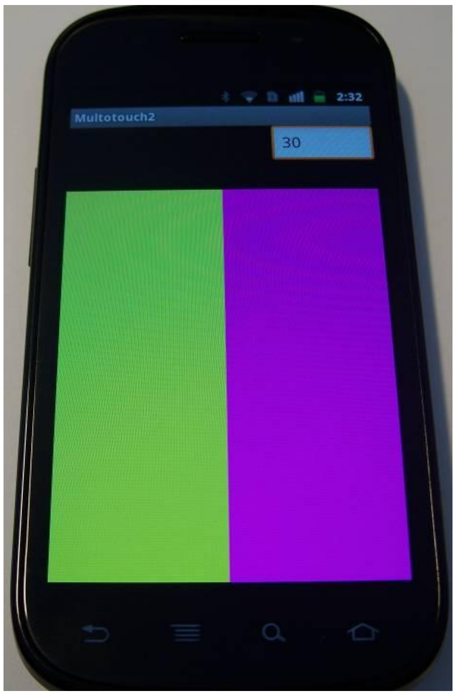


Figure 12, Second page multi-touch

.xml code:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
            xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_height="fill_parent"
   android:layout_width="fill_parent"
   android:id="@+id/MainLay">

   <LinearLayout
                        xmlns:android="http://schemas.android.com/apk/res/android"
                        android:orientation="horizontal"
            android:layout_height="fill_parent"
            android:layout_width="fill_parent">

            <TextView
                        android:layout_width="match_parent"
                        android:textAppearance="?android:attr/textAppearanceSmall"
                        android:text="@string/LeftCount"
                                    android:layout_weight="1"
```

```
                android:id="@+id/LeftCount"
                android:layout_height="wrap_content"
                android:layout_alignParentLeft="true"
                android:layout_marginTop="0dip"
        ></TextView>

        <TextView
                android:layout_width="match_parent"
                android:textAppearance="?android:attr/textAppearanceSmall"
                android:text="@string/RightCount"
                android:id="@+id/RightCount"
                android:layout_height="wrap_content"
                android:layout_marginTop="0dip"
                android:layout_weight="1"

        ></TextView>

        <EditText
                android:id="@+id/TimeDown"
                android:layout_width="fill_parent"
                android:layout_height="wrap_content"
                android:layout_alignParentRight="true"
                android:layout_weight="1"
                android:editable="true"
                android:text="30"
                android:clickable="true"
                android:enabled="true"
                android:numeric="integer"
                        android:cursorVisible="false"
        ></EditText>
</LinearLayout>

<TableLayout
xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:orientation="vertical"
  android:gravity="center_horizontal"
  android:id="@+id/Table">
```

```
    <TableRow
    android:gravity="center_horizontal"
    android:layout_weight="1"
                android:layout_height="wrap_content"
                android:layout_width="fill_parent"
                android:id="@+id/Row">

      <LinearLayout
                      android:layout_marginTop="80dip"
      android:background="#FFFF00"
      android:id="@+id/Button1"
      android:layout_width="wrap_content"
      android:layout_height="match_parent"
      android:layout_weight="1"/>

      <LinearLayout
                      android:layout_marginTop="80dip"
      android:background="#FF00FF"
      android:id="@+id/Button2"
      android:layout_width="wrap_content"
      android:layout_height="match_parent"
      android:layout_weight="1"/>
    </TableRow>
  </TableLayout>
</RelativeLayout>
```

.java code:

```
package multitouch.com;
import android.app.Activity;
import android.os.Bundle;
import android.os.Handler;
import android.view.MotionEvent;
import android.widget.TableLayout;
import android.widget.TextView;
import android.widget.TableRow;
import android.widget.LinearLayout;
import android.view.View.OnTouchListener;
import android.view.View;
```

```
public class Multotouch2Activity extends Activity {
        int i=0;
        protected TableLayout Table;
        protected LinearLayout Button1, Button2;
        protected TableRow Row;

protected String Button1text= new String();
protected String Button2text = new String();



private Handler handler = new Handler();

/** Called when the activity is first created. */
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);
    setViews();
    handler.postDelayed(mUpdateUITimerTask, 1);
}

private final Runnable mUpdateUITimerTask = new Runnable() {
    public void run() {
        updateTextField();
    }
};

private void updateTextField() {
    TextView LeftCount = (TextView) this.findViewById(R.id.LeftCount);
    TextView RightCount = (TextView) this.findViewById(R.id.RightCount);

    //view1.append(sqText + "\n");
    LeftCount.setText(Button1text);
    RightCount.setText(Button2text);

    handler.post(mUpdateUITimerTask);
}
```

```
private void setViews() {
            Button1 = (LinearLayout) this.findViewById(R.id.Button1);
            Button2 = (LinearLayout) this.findViewById(R.id.Button2);
            Row = (TableRow) this.findViewById(R.id.Row);

            Button1.setOnTouchListener(new TouchListener());
            Button2.setOnTouchListener(new TouchListener());
            Row.setOnTouchListener(new TouchListener());
}

private class TouchListener implements OnTouchListener
{
   @Override
   public boolean onTouch(View v, MotionEvent event)
       {
          if(event.getPointerCount()>2) {
            Button2text = "GOT THREE POINTERS!!";
          Button1text = "GOT THREE POINTERS!!";
       }
       if(event.getAction() == MotionEvent.ACTION_DOWN)
       {
                      switch(v.getId())
                      {
                                 case R.id.Button1:
                                            Button1text = "Button1 Pressed "+ event.getX();
                                            break;
                                 case R.id.Button2:
                                            Button2text = "Button2 Pressed"+ event.getX();
                                            break;
                      }

       }
       else if(event.getAction() == MotionEvent.ACTION_POINTER_DOWN ||
                 event.getAction() == MotionEvent.ACTION_POINTER_UP)
       {
          switch(v.getId())
          {
            case R.id.Button1:
                Button1text = "Button1 Pointer Down::Pointer count = "
+String.valueOf(event.getPointerCount())+"\n";
               break;
               case R.id.Button2:
                Button2text =  "Button2 Pointer Down::Pointer count = " +
String.valueOf(event.getPointerCount()) + "\n";
               break;
             }
           }
           else if(event.getAction() == MotionEvent.ACTION_POINTER_1_DOWN)
           {
             switch(v.getId())
             {
               case R.id.Button1:
                Button1text = "Button1 Pointer1Down::Pointer count = " +
String.valueOf(event.getPointerCount()) + "\n";
               break;
               case R.id.Button2:
                Button2text =  "Button2 Pointer1Down::Pointer count = " +
String.valueOf(event.getPointerCount()) + "\n";
               break;
             }
           }
           else if(event.getAction() == MotionEvent.ACTION_POINTER_2_DOWN)
           {
             switch(v.getId())
             {
               case R.id.Button1:
                Button1text = "Button1 Pointer2Down::Pointer count = " +
String.valueOf(event.getPointerCount())+ "\n";
               break;
               case R.id.Button2:
                Button2text =  "Button2 Pointer2Down::Pointer count = " +
String.valueOf(event.getPointerCount()) + "\n";
               break;
             }
           }
           else if(event.getAction() == MotionEvent.ACTION_POINTER_3_DOWN)
           {
            switch(v.getId())
            {
             case R.id.Button1:
               Button1text =  "Button1 Pointer3Down::Pointer count = " +
String.valueOf(event.getPointerCount()) + "\n";
             break;
             case R.id.Button2:
               Button2text =  "Button2 Pointer3Down::Pointer count = " +
String.valueOf(event.getPointerCount()) + "\n";
             break;
           }
         }
         else if(event.getAction() == MotionEvent.ACTION_UP)
         {
            switch(v.getId())
            {
             case R.id.Button1:
               Button1text = "Button1 UP::Pointer count = " + String.valueOf(event.getPointerCount()) +
"\n";
             break;
             case R.id.Button2:
               Button2text =  "Button2 UP::Pointer count = " + String.valueOf(event.getPointerCount()) +
"\n";
             break;
           }
         }
        return true;
    }

}}
```

Common to all the windows are two more files, the strings.xml and the project manifest. These two files are shown below.

Strings.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello World, ProjectActivity!</string>
    <string name="app_name">Project</string>
    <string name="LeftButton">Left</string>
    <string name="RightButton">Right</string>
    <string name="LeftCount"> Left Count:0 </string>
    <string name="CountLeft">0</string>
    <string name="RightCount">  Right Count:0 </string>
    <string name="CountRight">0</string>
    <string name="Time">"    Time"</string>
    <string name="TimeDown">0</string>
    <string name="PatientsName">&quot;Type the Name and Surname of the patient&quot;</string>
    <string name="PatientsNameText">Name</string>
    <string name="DiagnosePatient">"Check box if patient has been previously diagnose with
Parkinson's Disease"</string>
    <string name="PatientsSex">&quot;Patient's Sex&quot;</string>
    <string name="PatientsSexSpinner">Select</string>
    <string name="DiagnoseCheck">Tap to check</string>
    <string name="ContinueButton">Continue</string>
    <string name="CancelButton">Cancel</string>

    <string-array name="PatientsSexSpinnerArray">
             <item>Select</item>
        <item>Male</item>
        <item>Female</item>
    </string-array>
    <string name="DateOfBirth">Date of Birth</string>
    <string name="DateOfBirthH">ddmmyyyy</string>
    <string name="PatientsSurnameText">Surname</string>
</resources>
```

Project manifest:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
      package="com.Strath.Project"
      android:versionCode="1"
      android:versionName="1.0">
    <uses-sdk android:minSdkVersion="9" />

    <application android:icon="@drawable/icon" android:label="@string/app_name"
android:debuggable="true">
        <activity android:name=".ProjectActivity"
```

```
            android:label="@string/app_name">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
                <activity android:name=".ProjectActivity1"></activity>
                <activity android:name=".popupfirstpage1"></activity>
                <activity android:name=".popupwindowTimeActivity"></activity>
  </application>
</manifest>
```

6. Application specifications

This section aims to set the specifications for the application and to emphasize some potential problems that can be found during the development of the application.

In specifying a QDG application there should be three main windows. The first window should be used by the person in charge of performing the test to enter subject information which might include the first name, the surname, date of birth and sex of the person to be examined. There must be a Continue button and a Cancel button to perform the test or to abort the test. A pop up screen should appear if any of the fields are empty or have not been changed when the Continue button is pressed. The personal details introduced in the fields should be used to create a unique SQL data base before changing to the second screen when the Continue button is pressed. The SQL data base should be encrypted and not contain the name, surname, date of birth or sex of the tested person, that information should be used to generate a unique code to identify the person each time that the subject is tested.

The second main screen should contain two test views to show the number of times that each of the buttons have been tapped. It should also have a text box where the person responsible for the test should introduce the time in seconds that the test should be run for. There must be one view with a drawing of two buttons filling the rest of the window. When the time has been set an audio cue should sound giving instructions to the person who is going to perform the test. The test should start immediately after the first button has been hit. The instruction to the subject is to simply perform alternating finger taps to the targets as fast as possible for the duration of the test. When a button is pressed its side of the window should change colour and stay with that colour until it is released. A different note for each button should also be played every time that a button is hit. The test should run until the set time is up which can be signalled by an audio alarm. The following information should be gathered to populate in the background the SQL data base created in the first screen. Each tap of the screen should generate the following information:

- The time at which the pointer went down
- The location in the screen
- The pointer id
- The number of pixels been pressed

- The movement of the pointer while touching the screen
- The amount of pointers touching the screen
- The time at which the pointer releases the screen

The test time should be counted on the background and every so often updated on the text box of the main second window.

The third screen should show the information that has been gathered in the SQL data base in the form of graphs. Each graph generated from a subject that is possibly suffering from Parkinson’s disease or other motor function disorder should be contrasted with a graph generated from several people of the same age band that do not suffer from any known disease. There should also be the possibility of being able to contrast the graphs from previous tests. This would allow disease progression or effectiveness of treatment to be monitored.

6.1. Potential problems

There are some potential problems that can delay the development of the application. A SQL data base has to be created in the first page, then it has to be populated with data when performing the test in the second window. That information has to be accessed in the third window to produce the graphs. The SQL data base has to be stored in the memory card of the phone, taking processing time. This may cause a limitation of how much and how fast the data can be stored. Two tasks have to be performed in the background while the test is been carried out, thus using the multitasking capabilities of the smart phone. These two tasks are, counting the set time and storing the parameters retrieved from each tap in the window into the SQL data base.

The multi-touch capability of Android OS needs to be understood before developing the application as this can cause problems and delay the development of the application.

7. Discussion

Most of the medical applications found within the different markets have low ratings or no ratings by the public and many of them are for recreation. None of the searched applications have been tested or their results compared against the results obtain from a reliable medical device for this thesis. Not many of those applications have been given a CE mark or FDA approval. The results obtained in applications that have not been certified are not reliable.

Although the dual core mobile phones are powerful devices none of them have been yet approved to work as medical devices and the use of mobile phone communications within some areas in the hospitals is forbidden. Most of the mobile phones have the possibility of switching off the wireless communications making it possible for some applications to be used within the hospital environment. For example the iPhone 4 has the airplane mode capability that allows the wireless communications to be switched off. It is not possible to only switch off the GSM and leave on the Wi-Fi or Bluetooth systems that could be used within some areas of the hospital environment.

Some of the challenges are, to create an application that complies with the Medical Device Directive regulations.

There are hospitals that are introducing smart phones to increase their work efficiency. Those hospitals are only testing what could be improved by using the devices.

There are companies like Mubaloo or MSM Software that develop applications for other companies including Healthcare organizations.

One of the biggest expenses involved in the development of a medical application taken to market would be the conforming to the Medical Device Directives regulations. Regulations coupled with the necessity for medical trials to be performed would mean the application could take a significant amount of time to get into the market.

The mobile health market is expected to grow during the next years to reach $1.7 B by 2014 [28]. McKinsey & Company speculate that the global market for mobile health could worth up to $60 billion [29].

Areas of the world where medical services are limited could benefit from mobile health care as it would provide easier and more flexible access to knowledge and advice [29].

An area that has not been yet saturated with medical applications is related to motor functions. This area could use only the smart phone sensors and communication systems to help in the diagnosis follow up treatments and by monitoring the day to day activities of the individuals suspected of having motor function problems as well as to remind them to take their medication. The sensors required for this type of application are, a touch screen, accelerometer and gyroscope and a loud speaker as an actuator. The information could then be sent via Bluetooth, Wi-Fi or 3G to the clinician or to a carer.

The medical application for QDG was started and the graphics for the first window have been developed, a popup error message comes up if one of the fields have not been filled when the Continue button is pressed. If all of the fields have been filled, the second window opens when the Continue button is pressed. There were some problems encountered when developing the first window and the error window. At this time the error window is not a popup window as it was more straightforward to develop an activity rather than a popup window. At this time no SQL database is created when the Continue button is pressed due to the time spent in developing the second window. The graphics shown in figure 11 are two text views, a text box and two buttons representing the two target areas for the subject to aim at. Currently, each time a button is pressed a text view is updated with the number of times that the corresponding button has been touched. The text box shows a number that represents the time in seconds that the test should be run for. At this time this function is not operational. The issue with Android OS was that it does not support multi touch within different views. When one button is pressed the other views are locked resulting in the second finger not being detected if the first finger is already in contact with the screen. To resolve this issue the graphics and code for figure 12 were tested. Each time that a finger touches one of the coloured target areas the exact horizontal position is shown in one of the text views. If the finger moves while touching the screen the text view is updated specifying this movement and if the finger leaves the screen then text view is updated and shows that the pointer is up. If one finger is already on the screen and the second finger hits the other target area of the screen the text view related to the first target area to be touched gets updated showing the number of pointers which are touching the screen. Nothing happens until the first finger is released from the screen. If the first finger released from the screen before the second finger, the text view shows incorrect information. This is what was being tested and this code needs to be developed further in more detail.

The next stage of the programming would be to finish the second window by finishing the multi touch function and develop a timer function that works in the background. This would be followed by the creation of the SQL database in the first window and the challenge of recalling it on the second window and using the data to create the graphs to be shown in the third window. An in-depth knowledge of Java programming would aid the development of the application further.

8. Citations:

1. *Gartner Says Worldwide Mobile Device Sales to End Users Reached 1.6 Billion Units in 2010; Smartphone Sales Grew 72 Percent in 2010*, in *Egham* 2011, Editor. Gartner: http://www.gartner.com/it/page.jsp?id=1543014.
2. *Smartphone*. 2011, Wikipedia: http://en.wikipedia.org/wiki/Smartphone
3. Gartner, *Gartner Says Android to Command Nearly Half of Worldwide Smartphone Operating System Market by Year-End 2012.* 2011 http://www.gartner.com/it/page.jsp?id=1622614.
4. AppBrain, *Number of available Android applications.* 2011 http://www.appbrain.com/stats/number-of-android-apps.
5. Viticci, F., *How Many iPhone Apps Are There? 306,554 – And 60,000 iPad Apps* 2011 http://www.macstories.net/news/how-many-iphone-apps-are-there-306554-and-60000-ipad-apps/.
6. Zephyr, *Zephyr BioHarness™ BT.* 2010 http://www.zephyr-technology.com/wp-content/uploads/2010/12/BioHarness-BT-Data-Sheet-2010-11-18.pdf.
7. Zephyr, *BioHarness™ BT.* 2010 http://www.zephyr-technology.com/store/bioharnessbt.html.
8. Withings, http://www.withings.co.uk/en/index/?gclid=CIH55tLGmqoCFUoc4Qod0Q0Uyg.
9. Wikipedia, *Android (operating system).* 2011 http://en.wikipedia.org/wiki/Android_%28operating_system%29.
10. Android, *Android developers.* http://developer.android.com/index.html.
11. Jorsher, *How to install Android SDK v1.0 - Step by Step* 31/10/2008 http://androidcommunity.com/forums/f4/how-to-install-android-sdk-v1-0-step-by-step-5749/.
12. Trends, O.M., *Smartphone market share, by country.* 2011 http://www.onlinemarketing-trends.com/2011/04/smartphone-marketshare-2011-italyus-aus.html.
13. Wikipedia, *iOS.* 2011 http://en.wikipedia.org/wiki/IOS_%28Apple%29.
14. Inc, A., *iOS Developer University Program.* 2011 http://developer.apple.com/programs/ios/university/.

15. Wikipedia, *Windows Phone 7.* 2011 http://en.wikipedia.org/wiki/Windows_Phone_7.

16. Microsoft, *Windows Mobile 6.5.3 Developer Tool Kit.* 2011 http://www.microsoft.com/download/en/details.aspx?displaylang=en&id=5389.

17. Qualcomm, *Product Details.* 2011 http://www.qualcomm.com/snapdragon/specs.

18. Samsung, *Exynos 4210.* 2011 http://www.samsung.com/global/business/semiconductor/productInfo.do?fmly_id=844&partnum=Exynos%204210&xFmly_id=229.

19. Samsung, *S5PC110.* 2011 http://www.samsung.com/global/business/semiconductor/productInfo.do?fmly_id=834&partnum=S5PC110&xFmly_id=229.

20. Nvidia, *Nvidia Tegra 2.* 2011 http://www.nvidia.com/object/tegra-2.html.

21. *Touchscreen.* 2011 http://en.wikipedia.org/wiki/Touchscreen#Capacitive.

22. *Near field communication.* 2011 http://en.wikipedia.org/wiki/Near_Field_Communication.

23. Miller, M., *Top 10 smartphones of 2011 ... for now.* 2011 http://www.zdnet.com/blog/cell-phones/top-10-smartphones-of-2011-for-now/5683.

24. Ana Lisa Taylor Tavares, M., BS,1 Gregory S.X.E. Jefferis, PhD,2 Mandy Koop, MS,1, P. Bruce C. Hill, 1 Trevor Hastie, PhD,3 Gary Heit, MD, PhD,4, and M. and Helen M. Bronte-Stewart, MSE1,4*, *Quantitative Measurements of Alternating Finger Tapping in Parkinson's Disease Correlate With UPDRS Motor Disability and Reveal the Improvement in Fine Motor Control From Medication and Deep Brain Stimulation.* Movement Disorders, 2005. **20**(10): p. 1286–1298.

25. *Helen M. Bronte-Stewart, M., MD, †Long Ding, MS, *Corinne Alexander, BA, ‡Yi Zhou, PhD, and and P. *George P. Moore, *Quantitative Digitography (QDG): A Sensitive Measure of*
Digital Motor Control in Idiopathic Parkinson's Disease. Movement Disorders, 2000. **15**(1): p. 36-47.

26. Directive, M.D., *What is a Medical Device? .* 2007 http://www.mhra.gov.uk/Publications/Regulatoryguidance/Devices/Otherdevicesregulatoryguidance/Frequentlyaskedquestions/index.htm.

27. Directive, M.D., *How do I CE mark my medical device? .* 2007

http://www.mhra.gov.uk/Publications/Regulatoryguidance/Devices/Otherdevicesregulatoryguidance/Frequentlyaskedquestions/index.htm.

28. Versel, N., *Enterprise m-health will be $1.7B market by 2014.* 2010 http://www.fiercemobilehealthcare.com/story/enterprise-m-health-will-be-17b-market-2014/2010-11-23.

29. Versel, N., *Worldwide m-health market opportunity could reach $60 billion.* 2010 http://www.fiercemobilehealthcare.com/story/worldwide-m-health-market-opportunity-could-reach-60-billion/2010-02-23.

9. Bibliography

Burnette, E (2010) *Hello Android; Introducing Google's Mobile Development Platform* (3rd ed.), Pragmatic Programmer, LLC.

Gagenta, M (2011) *Building Application for the Android Market, Learning Android*, O'Reilly Media, Inc.