Recent Advances in Ambient Intelligence and Context-Aware Computing

Kevin Curran
University of Ulster, UK

A volume in the Advances in Computational Intelligence and Robotics (ACIR) Book Series

Information Science REFERENCE
An Imprint of IGI Global

Managing Director:	Lindsay Johnston
Managing Editor:	Austin DeMarco
Director of Intellectual Property & Contracts:	Jan Travers
Acquisitions Editor:	Kayla Wolfe
Production Editor:	Christina Henning
Development Editor:	Erin O'Dea
Typesetter:	Amanda Smith
Cover Design:	Jason Mull

Published in the United States of America by
Information Science Reference (an imprint of IGI Global)
701 E. Chocolate Avenue
Hershey PA, USA 17033
Tel: 717-533-8845
Fax: 717-533-8661
E-mail: cust@igi-global.com
Web site: http://www.igi-global.com

Copyright © 2015 by IGI Global. All rights reserved. No part of this publication may be reproduced, stored or distributed in any form or by any means, electronic or mechanical, including photocopying, without written permission from the publisher. Product or company names used in this set are for identification purposes only. Inclusion of the names of the products or companies does not indicate a claim of ownership by IGI Global of the trademark or registered trademark.

Library of Congress Cataloging-in-Publication Data

Recent advances in ambient intelligence and context-aware computing / Kevin Curran, editor.
pages cm
Includes bibliographical references and index.
ISBN 978-1-4666-7284-0 (hardcover) -- ISBN 978-1-4666-7285-7 (ebook) -- ISBN 978-1-4666-7287-1 (print & perpetual access) 1. Ambient intelligence. 2. Context-aware computing. I. Curran, Kevin, editor.
QA76.9.A48R43 2015
004.01'9--dc23
2014036868

This book is published in the IGI Global book series Advances in Computational Intelligence and Robotics (ACIR) (ISSN: 2327-0411; eISSN: 2327-042X)

British Cataloguing in Publication Data
A Cataloguing in Publication record for this book is available from the British Library.

All work contributed to this book is new, previously-unpublished material. The views expressed in this book are those of the authors, but not necessarily of the publisher.

For electronic access to this publication, please contact: eresources@igi-global.com.

Chapter 6
Common Problems Faced When Developing Applications for Mobile Devices

Kevin Curran
University of Ulster, UK

Sean Carlin
University of Ulster, UK

Joseph McMahon
University of Ulster, UK

ABSTRACT

Mobile application development is relatively new and has seen growth of late. With this rapid expansion, there are growing pains within industry, as the usual time given to the evolution of an industry to learn from past mistakes has been significantly shortened and is even going on within the currently saturated market. Because of this, inexperienced developers are attempting to design applications based on what is of yet a shady set of design principals. This is providing problems during the development process and can be seen to be stifling innovation, as many developers have yet to get a grasp on the shift between traditional software engineering and what it means to implicate these designs on a mobile device. This chapter analyses these difficulties in depth, as well as attempting to draw solutions to these problems based on development in the context of the characteristics of mobile devices.

INTRODUCTION

Mobile Computing is currently growing and has no plans of slowing down in the near future. Gartner states that worldwide sales of mobile devices to end users totaled 428.7 million units in the second quarter of 2011, a 16.5 percent increase from the second quarter of 2010 (Pettey & Goasduff, 2011). This growth is attracting more and more developers to move away from traditional applications and web development. Many of these developers are now focused on developing applications for mobile devices. A mobile device is a generic term used to refer to a variety of devices that allow people

DOI: 10.4018/978-1-4666-7284-0.ch006

Copyright © 2015, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

to access data and information from where ever they are. This includes cell phones and portable devices (Hakoama & Hakoyama, 2011). A mobile device is a very broad term in the fact that it can refer to laptops, net books, smart phones and the most recent form of mobile technology known as tablets. When a person discusses mobile devices today, they are generally referring to a smart phone or tablet. Developers whether they are new to mobile development or seasoned veterans, run into many problems and are required to make tough calculated decisions based on number of factors affecting the deployment/ accessibility of their mobile application.

A large portion of Smartphone sales are driven by the availability of apps, and as the demand for apps increases, so grows the demand for app developers and quality applications. The fast pace at which the technology is being incorporated into mobiles and the pace at which it is developing brings with it its own problems. Since mobile application development is a relatively new field in computing and is also becoming a very profitable and enterprising market, several changes in the development process have to be taken onboard in order to create a successful product, such as how the application is going to perform from device to device, something that rarely had to be considered with developing software for desktop computers. While Smartphones now usually come pre-installed with applications which carry out a wide range of functionality specific to that operating system and device, utilizing the specific hardware in that circumstance, custom applications can also be downloaded onto the device and will vary in usefulness depending on their compatibility with that device's hardware and software. Developers must recognise what customers want from their devices, and make their product stand out in what is becoming a very over-saturated market, which in itself is one of the biggest challenges faced when developing mobile applications. As the number of technologies around us increase and start to pervade our everyday routines, our methods of interacting with these technologies is also changing. We are seeing an increase in the number of wireless networks in public areas, as well as technologies that are interacting with each other wirelessly or over the internet and the mobile phone market has evolved naturally to incorporate and interact with these technologies. Social networking has emerged as one of the biggest trends in recent years, and has revolutionised the way in which people communicate with each other. This has converged with the evolution of mobile technology, providing users with the convenience of having access to the wider world at their fingertips. This is a perfect example of how the functionality of mobile devices has evolved to meet user's needs and has created a large demand from the users of mobile devices for more and more applications which incorporate these technologies. As the speed of mobile devices and the constant development of new technologies become available, more devices are being released with different sets of features to provide for a range of user needs. This has created a very fragmented development community, as the wide range of platforms and devices available mean that development cannot be too focused on a single set-up if it is to reach a wide audience or make money. Applications have been developed for many different tasks. Originally conceived as software to increase efficiency and productivity, such as personal planners on devices such as PDAs, as the hardware evolved and the widespread use of mobile devices continued, so did the variety of apps becoming available, with these mobile devices offering alternative platforms for casual gaming, offering businesses a portable way of communicating data and information to each other, as well as extensive use in the medical and educational fields. The portability and increasing capabilities of Smartphones and PDAs intrigued developers into incorporating media which we normally associate with desktop computers into the handhelds, providing users another avenue to access their emails, social networks, maps etc.

while not at the desk, as well as seeing many innovations using the location based technologies of the mobile devices.

So far, amongst the development community, no widely agreed on development model has been presented and thus the methods used in mobile application development are ultimately unwieldy as of yet. In this chapter, the problems faced when developing applications for mobile devices will be highlighted and evaluated, with a critical analysis given to any techniques which may be working in this field and any techniques which are benefitting or may be employed in the future. Developers have to deal with a slight paradigm shift in the development process. One of such main differences and challenges is the fact that the traditional PC development process doesn't translate very well to development on mobile platforms. Because of the wide range of mobile platforms and hardware available, portability between various technologies is a major issue in mobile app development, as each user may have a different mobile device which may function differently. This is not usually an issue with developing applications for computers, as development is usually very platform specific, therefore it would only have to be developed with that platforms functionality and specs in mind, with testing being able to be automated. Because of the vast amount of platforms out there, some of the developers of these platforms have released development guidelines based on developing on their specific devices. While this may be useful for developing applications to those specific platforms specifications, there are very little agreed on guidelines for developing over many platforms. With tools available to making cross-platform coding easier, there still lies the problem of incompatibilities that may crop up in the field. This means that a lot of mobile application development relies solely on the developer if they are to make the product compatible over different devices. This is the main issue when it comes to developing applications for mobile devices, as portability between devices has to be taken into account in the coding process. Due to the large amount of operating systems available, their respective IDE's used when developing them also have their own coding languages and specific criteria for being loaded onto a mobile device. This is a problem for developers, as a wide knowledge and code based knowledge is needed if an application is to be deployed on several platforms. This chapter provides some insight into many common problems developers must face and overcome to develop an application for mobile devices.

BACKGROUND

The developer must consider the various physical limitations they are faced with when developing a mobile application such as:

- **Resource Limitations:** Mobile devices have a fraction of resources available to them in comparison to today's modern computers. Every mobile device designed and developed has a difference in physical resources to the previous or next. The current mobile devices have less ram available for applications and their processors are much smaller and slower in relation to modern desktop computers. This creates a large problem for developers as they are required to be much more economical with their resources and thorough in their planning in order to create a high performance application which works on multiple devices with varying resources.
- **Screen Size Limitations:** Developers must be creative and find a more visually friendly way of displaying information as they have smaller screens averaging between 2.5 – 4.0 inches with a multitude of screen resolutions. This can be a daunting task and is not feasible to test every screen

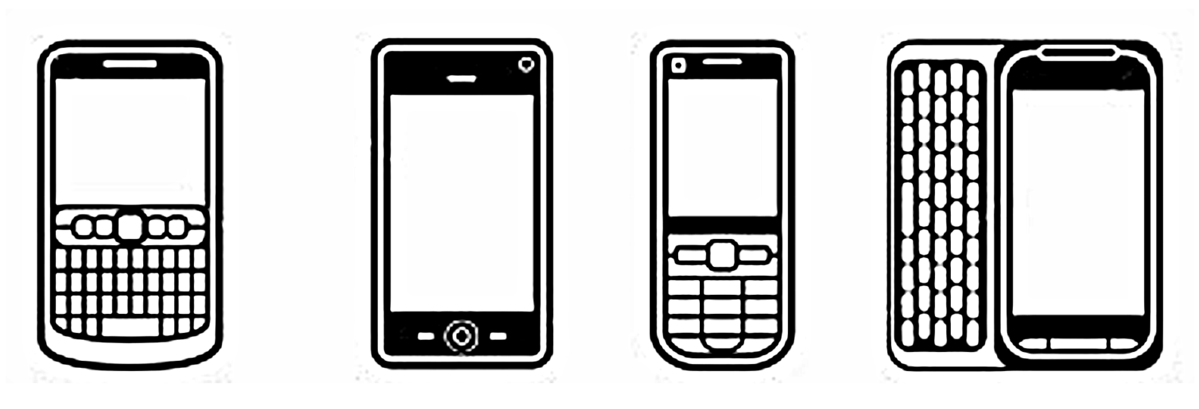
size and resolution, which is why the developer must be completely up to date with the industries best practices and standards.

- **Power Limitations:** Inefficient mobile applications tend to consume a lot of battery life due to the waste of resources through unreleased ram and processes. With mobile devices being battery powered they only have a limited power supply until they are required to be plugged in and recharged, some mobile devices having as little as one days battery life. This puts the developer under pressure to make the application as efficient as possible in order to keep battery life consumption to a minimum while maximizing the applications performance.
- **Multiple Form Factors:** Mobile devices come in a variety of shapes, sizes and hardware configurations. Some have physical keyboards or virtual keyboards; some have touch screens; some have high resolutions others have low; some can be large and clunky or small and slim; some have cameras. A visual representation of different types of form factors of mobile devices can be seen in Figure 1. This kind of variety in form factors and hardware configurations is a very real problem for developers and can be a large factor affecting the usability of an application on a mobile device.

The fact that Smartphones are also mobile, incorporating various location based technologies, brings along a whole new set of design challenges as well. There are multiple dimensions of mobility that have to be considered during the mobile development process (B'Far, 2005). These dimensions are location awareness, network connectivity quality of service, limited device capabilities (storage and CPU), limited power supply, support for a wide variety of user interfaces, platform proliferation, and active transactions. These dimensions in relation to the development process mean that developers will have to take a different approach to the design of their applications, and begs a new set of design principals. This is a challenge as some of these dimensions may present many limitations on the intended functionality of the application, whilst some present completely new opportunities, opening a whole new range of technologies which can be implemented into, but may complicate the plan of the project. This also strongly affects the testing of the application.

Testing is a major process in the development lifecycle, and it has to be thoroughly executed if the final product is to function as intended in the field. In developing for mobile devices, testing is more difficult as there are many more variables to consider, as well as a lack of proper test cases by which to run the applications against. Because of the inherent variations in mobile devices, there

Figure 1. Showing an example of different form factors of mobile devices



is a range of different functionality and capabilities with each device and operating system, as well as the characteristics of being mobile as described earlier. This means that field testing would be impossible on every device, because there are simply too many different devices, so the scope of testing would have to be narrowed and specialized. This requires extra planning and time on the developer's part, which can put a lot of stress on the project. Emulators can be used to simulate a lot of the applications functionality over a variety of devices without having to physically have those devices at hand. This can only test the application to a certain extent though, as the emulator may not cover a lot of different models, as well as new ones constantly being released and a different emulator would have to be used for each different operating system. Emulators would only go so far in testing, as it would not be able to simulate a live environment as well as having the application running on a device. Things that vary in real life situations, such as signal strength, battery life remaining, and memory available would usually be overlooked, and many of the test cases may not be specific enough to cover many of these facets, possibly causing problems when the software is deployed in the field. Automation is a technique widely used for testing in the development industry. By formalizing the approach to testing by using software to carry out a set of test cases on the product, testing can be sped up and can determine whether the product functions as expected. When it comes to developing mobile applications, the number of variables that have to be taken into account, such as the constraints of the mobile platform itself, make pinning down test cases, and as a result, automation difficult, meaning that the developer will have to carry out most of these manually, costing time and resources (Kumar, 2007).

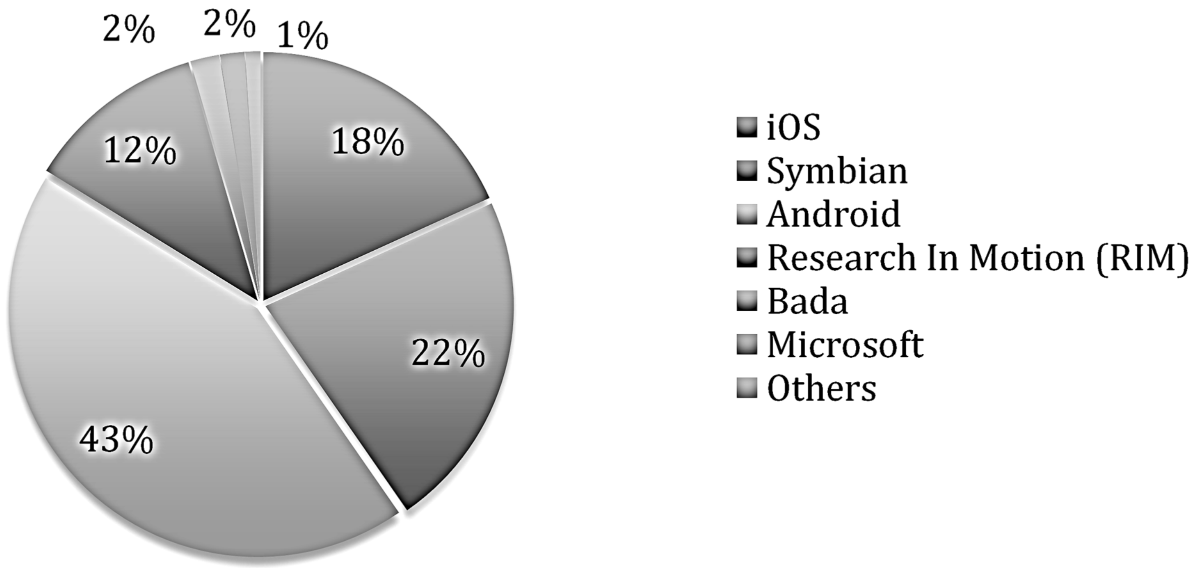
The portability of the application between hardware and operating systems will be one of the main factors that will complicate the process when it comes to mobile app development. Inconsistencies between functionality of devices mean that the scope of a project will have to be very defined if it is to work on a wide range of devices. Factors that will have to be taken into account are the device's display, memory and the buttons, touch capabilities and sensors on the device. Because of these inherent inconsistencies of mobile devices in general, portability between platforms will have to be meticulously planned beforehand, and may involve lots of re-coding and re-development during the course of the project, with many problems that may crop up during the development phase which could not have been predicted beforehand, and could cause significant setbacks in the project due to incompatibility of the software between platforms. Aspects such as screen size and resolution are one of the main variations between models of mobile devices. If an application cannot stretch to fit the size of a screen, or the screen can't display some of the colours or match the resolutions, errors may crop up or crash the application altogether, which means precise coding will have to be incorporated into the application in order to detect the device's specifications and format the display as to show it correctly on the device's screen. Many devices also have the capability of turning the device to create a landscape view of the screen. Again, this will change the resolution of the screen. This is not a problem commonly associated with traditional software development and with the constant proliferation of mobile devices means that this is a challenge for developers to be able to predict and plan what methods they are going to use to be able to port their application over various device's displays, as well as if the application is being displayed as landscape or scaled down. Some applications may also be very large in size, or take up a lot of RAM in order to run. This may be due to the complexity of the application, or if it incorporates elements such as 3D graphics, which would require the device to be sufficiently fast in order to run it. This means that older hardware may not be able to keep up with the demands of the app, causing crashes. This is difficult to predict and test for on the developer's part, as there can be differences between a device's total RAM/CPU and the devices available RAM/CPU, depending on how many applications are running on the background amongst other factors. Unlike traditional desktop software development, where this is not usually a constraint, mobile devices are still lacking in the area of computational power, which will mean having to code for any situations in which a lack of memory or any other factors may impede the application's normal running, costing more time on the developers part. Many mobile devices these days contain several sensors in order to detect light levels, orientation etc. This means that applications geared towards making use of these functions will have to be very specific in scope, with a lot of research done on the developer's part as to which devices contain the appropriate technology and which devices and operating systems they will be targeting. All these factors combined will provide a big challenge on the developer's part, as much coding which could prepare the application for situations, such as dealing with a hardware or software incompatibility may also slow it down, or consume more resources if the application's code becomes too bloated with exception handling, taking longer to process at runtime.

OPERATING SYSTEM CONSIDERATIONS

Mobile applications cannot be built without an operating system to run on therefore one of the most fundamental requirements for a developer building an application for a mobile device is to choose which operating systems they support. A key factor in influencing which operating system to develop their application for is how many people can access their application. Figure 2 displays the market shares of today's current mobile operating systems (Pettey & Goasduff, 2011).

The higher the user base that can access the application usually materializes to more profit/ downloads for the application developer. Judging by the results above this is currently Android, which holds a 43% of the worldwide market share in mobile devices. This is an important factor

Figure 2. Mobile device operating systems market share % August 2011



for the developer but it is not the only factor, the developer must think of other implications when choosing an operating system such as:

- **Privileges:** If the developer wishes to access the hardware of the mobile device and manipulate it to do exactly what they require, they will need root permissions. This is not a feature all operating system providers (namely apple) offer for their mobile devices, developers will not be able to access low-number network ports on iOS, for example, or do anything else that would typically require root or administrative access on a desktop computer (Mark, Nutting & LaMarche, 2011). This is a serious limitation when it comes to the developer's creativity and originality in their application. In contrast the more open android operating system developed by Google boasts "each application's process, data storage, and files are private unless explicitly shared with other applications via a full permission-based security mechanism" and allows developers to access any of the devices hardware components (Meier, 2010).
- **Developer Costs and Prerequisites:** When a developer chooses to write an application there will always need to be an investment of time, but in some cases money is also mandatory. An example of this is developing an application for iOS, the developer must register with apple as a developer, which costs £99 per year and they must also own a Macintosh computer to use the SDK. This can be an issue and hindrance for developers if they do not want to pay a yearly licensing fee or do not own a Macintosh computer.
- **Publishing Application Process:** Many of the mobile devices operating system providers offer application stores or market places for developers to publish and monetize their applications. These application stores allow the end users to purchase, download and run the application on their mobile device. Some of the popular application stores and market places today are Apples app store, Google's android marketplace, Microsoft's windows phone market place, Nokia's ovi store and Blackberry's app world. The application stores have their own advantages and disadvantages for the developer. The application stores vary in strictness and standards, for example Google's android market place has no approval process for application distribution (Meier, 2010). This caters for the developers because there are no real standards and they have complete control of how they wish their application to look, feel and function. Having no approval process can cause an inconsistency in application quality and leave the store bundled with poor and insecure applications to the annoyance of the common users who use the applications.

On the other hand Apples app store includes a robust user inter- face tool that enables developers to use prebuilt components, supported with detailed Human Interface Guidelines (or HIG) of how to use them, similar to a pattern library (Fling, 2009). Apple is strict and the application will be refused unless it meets their visual and resource guidelines. This has real benefits for the end user as they will have access to applications with a similar design and the store will have a certain standard and quality. But this can also be an issue for developers as strict guidelines can lead to application refusal, longer development times and costs.

WEB BASED, NATIVE, OR HYBRID APPLICATION DEVELOPMENT

Developers have to understand their target markets and what functionality of the mobile device they will need to access. Central to every development platform is its software development kit (SDK), which enables third party developers to deliver applications running on the plat-form (Holzer & Ondrus, 2009). To use the most core functionality of the mobile device such as the camera, SMS or phone, a native application must be built using an SDK, which supports the mobile device and operating system. Native applications can utilize the full power of the mobile device but can cut revenue for the developer as the application may not work on a different device running a different operating system, an example of this is developing an application using the android SDK and not being able to run the application on a windows phone, blackberry or iPhone without developing a standalone native application for each of these devices. If a developer decides to create an application that does not require the use of GPS, camera, accelerometer, access to file system or any core functionalities of the device then the developer can develop a web-based application. The combination of familiar web technologies with minimal native code enables you to develop your application once and deploy to a variety of mobile devices (Jacobs, 2011). Mobile web applications run through the mobile devices browser enabling it to run on a variety of devices and operating systems. This can free up a lot of development time and testing but the developer has to sacrifice the core functionalities of the mobile device and the device must also be constantly connected to the Internet.

The alternative to the native and web-based applications mentioned above is that developers may choose to develop a hybrid application, this is now possible due to some "cross-platform development tools, such as RhoMobile's Rhodes and the open source PhoneGap, which can be used to create native applications on various brands of Smartphones. Along the same lines, Netbiscuits, Appcelerator, Kyte, and other companies provides tools and frameworks to support the creation of mobile web and hybrid sites using their SDK or one of the previously mentioned environments" (Wasserman, 2008). This allows the developer to create small-medium sized applications that will be cross platform, this approach is generally adopted by developers who need to create an application on multiple platforms as fast as possible. It does require the developer to adhere to many core software development principles such as abstraction and the developer has to plan and design their application thoroughly.

PLANNING AND MANAGEMENT

When developing a mobile application it is no different than developing a traditional computer application in the fact that the developer has to meticulous in their planning. The developer must plan for the worst and hope for the best as anything can happen during the development process and it is important that the developer has some sort of contingency plan. This way the developer will be prepared for over running deadlines or underestimating how much the project would cost etc. Documentation is key when developing any kind of application as it will allow one developer to pick up where another has left off, it also allows more developers to join the project and get up to speed quickly which limits the risk of project failure. Planning the development of the mobile device's application is essential for its success.

A common problem when developing mobile applications is deciding how the application will make profit to give the developer a return in their investment. The developer is faced with a variety of business models and revenues streams and it can be difficult to assess each and choose the correct one for the application. It is important that the developer understands each revenue stream and business model described below and chooses to use the best models that compliment their application.

- **Paid Applications:** The developer would set a standard price for their application on the application store and the user would pay the one off fee that gives them unlimited access to the applications functionality. This is the most basic type of revenue stream for developers often the company which owns the store where the application is published will usually take a 30% cut of the apps revenue.
- **Subscription and Pay-Per-Issue Applications:** This is a model where the developer can get constant revenue by supplying the user with new issues or content on a regular bases. The user pays a monthly fee or for the new issue of content, this does however require the developer to constantly update their application with fresh content to justify the constant bills on the users.
- **Free Applications with Advertising/ Affiliate Marketing:** The developer could make a return on their investment by making the application free to download and use in-app advertising or affiliate marketing to produce revenue. Affiliate Marketing is marketing products or services for other companies in exchange for a commission (Volk, 2010). This marketing technique is massive in the web as a revenue stream and is quickly being introduced to today's mobile device applications. In-app advertising and affiliate marketing operates very similarly to the pay per click model of the web; this has been modified for the current mobile markets as a pay per action model. Developers must not ignore these revenues streams, especially if their application is free.

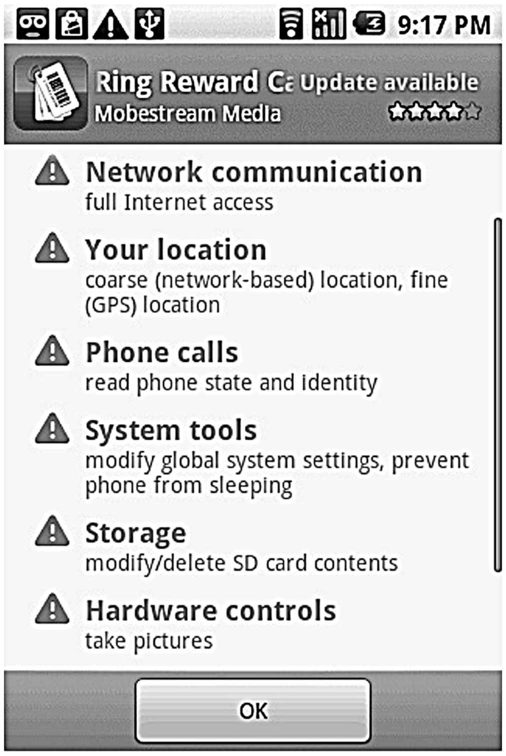
SECURITY CONCERNS

One of the major problems when developing a mobile application is how to make the application secure (Lowe, 2011; Kumar, 2007). The fact that mobile technology is relatively new and beginning to incorporate technologies and protocols from desktop computing, such as operating systems, IP addresses and their ability to send and download software and execute code means that it is open to all the vulnerabilities seen on desktop computers, and possibly many more. Due to the fact that mobile technology is mobile as well, opens the devices and its users up for a whole new range of possible compromise, due to the fact that mobile devices often contain location based functionality, spyware can take a lot more information from the devices than could traditionally be attained from computers. Viruses of completely new classes are also popping up, using SMS and Bluetooth connections as a whole new avenue to spread from device. Preventing these issues from affecting the application or what information the app has stored on the device will be problematic for the developer from a technical standpoint. As viruses are constantly evolving, with new methods of attack being constantly created, it is important that the application is kept as simple as possible, so that every possible area in which it can be attacked can be barricaded. This may require a lot of time on the developer's part, and could warrant the deployment of later patches post-release if the device is at significant risk due to the application. It will also be the developer's responsibility to ensure that their application does not look or act suspicious to the user, as this could cause the application to get a bad reputation and harm the applications success.

Developers must be transparent with the end users and explain why they need access to certain permissions on the mobile device. It is also the developer's responsibility to use self-signed certificates, which developers can generate without anyone else's assistance or permission. One reason for code signing is to allow developers to update their application without creating complicated interfaces and permissions (Burns, 2008). Without self signed certificates the application could come from anyone, so it is important that the developers sign the application as their own, this will build confidence with the users and distributors (Onias-Kukkonen, 2003). When downloading an application, that platforms app market will often contain some information about the application, as well as what the application has access to, as can be seen in Figure 3.

While it is clearly defined what the application has access to, this can be often off-putting and daunting for first time users. For example, say if a simple application such as a calculator wants access to the device's GPS services, it can be quite unclear as to why this is needed exactly, and can feel like it is impeding on the user's privacy causing them to possibly avoid the app altogether,

Figure 3. Application permissions in Android



9:17 PM

Ring Reward C: Update available
Mobestream Media

Network communication
full Internet access

Your location
coarse (network-based) location, fine (GPS) location

Phone calls
read phone state and identity

System tools
modify global system settings, prevent phone from sleeping

Storage
modify/delete SD card contents

Hardware controls
take pictures

OK

which can hurt application development. One way around this is to use the 'least privilege model', functioning on the least amount of privileges that are needed, for example if the app does not need access to the camera, it should not grant itself access to the camera, as this could impede on the processing speed or the running of other applications (Dwivedi et al., 2010), which will also give the user a more succinct overview of what the application is doing, so it may not seem as suspicious. Getting the application signed is also highly recommended and can reassure the user that the application and code are coming from a trusted source.

A lot of user information is handled by is connected to or stored physically on the mobile device, therefore if the device is stolen, the third party can possibly gain access to this information, which could be worth more than the device or possibly put the user at harm. This calls for a radical rethinking in the way that we handle our personal and sensitive information on mobile devices. One method to tackle this is not to store any of this information locally on the device. If information is stored on the device, it is usually only a matter of time before access can be gained to it through hacking or other methods. If the information can be stored remotely, the owner of the device will have more control over who can access this data. This will mean that future applications will have to be designed with the assumption that physical access to the device and the data contained on it is possible to third parties, with access to the information done over a remote server or a computer cloud. This may in turn though, inconvenience a genuine user, who will have to bypass security protocols when they want to access their own information. User identification technology is emerging increasingly in mobile devices, with some devices being able to identify user by fingerprint or facial recognition, which may go a long way in authenticating the user more quickly. Tools such as 'Zap it' by iAnywhere also present intriguing solutions to the aforementioned problem. This software allows the content on a phone to be erased remotely if certain criteria are not fulfilled, such as the user not signing in to the server in a set window of time, which would be useful in the case of a stolen device. Also the developer can implement more short term measures to ensure every possible level of protection for the user's information, such as sanitizing inputs so that exploits of the app cannot occur, as well as ensuring any information that does need to be stored locally is encrypted.

In certain operating systems the developers must use initiative and truly understand the fundamentals of the languages they are using. For example developers must be weary of creating buffer overflows by accident or by someone exploiting their code when using languages such as C/ Objective C. Developers must be concerned when allowing their applications to communicate to servers. All data transmitted to and from the server should be securely encrypted using SSL certificates especially when the data is of a sensitive nature such as medical and financial records. Once a developer publishes an application the application is never truly finished as bugs surface and loopholes become apparent, this is an inherent problem for developers when creating applications. Developers have to patch their applications and inform the users that their current version is out of date to keep their application secure.

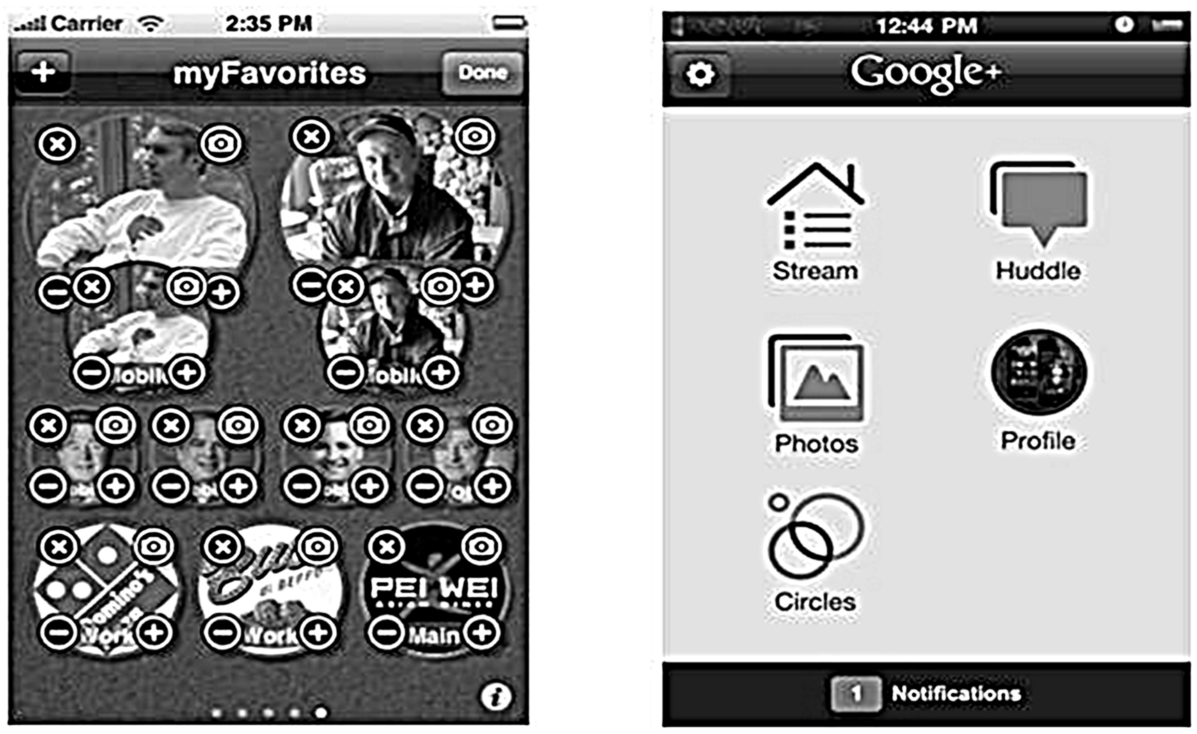
As mobile devices continue to bring together various forms of media and technology into one device, their users' information for these various applications is also stored on the device, or accessed from the internet via the device. This poses a security risk, with either the loss of the device containing some unsecured or unencrypted personal information potentially being stolen or even gaps in the code of an application which handles sensitive information being hacked and stolen. When developing an application that handles or stores personal user information or other sensitive information, it is important on the developer's part to decide how to handle this information in the safest possible way. From simple things such as displaying a "*" to hide the characters on the screen when a password is entered by the user, to encrypting any passwords or important information that is stored on the device, it is vital that the user feels that their details are secure, in order for the application to function successfully. The main problem with developing with this in mind is that, because the concept of carrying out transactions or storing important information using mobile devices is relatively new, and they can be used for such a range of different media, with users usually using their mobile device to converge all of their data, such as email, social network profiles, notes, messages, pictures, videos etc. into one, a single device can contain access, restricted or not, to a lot of that person's information. If someone gets a hold of the physical device it is only a matter of time before the security constraints are broke, and whatever data contained on the device accessed. This will pose a challenge to the developer, as it will mean adjusting their design principles around how information is going to be handled and protected on the system – this can range from using a dedicated server to cloud computing in order to allow the user to access their information, or to allow the system to verify the user (Dwivedi et al., 2010). While it is up to the user's discretion and common sense what information they choose to use their mobile device to handle, it is also the developer's responsibility, if the user is using their application, to ensure that their content is protected. Due to the need for applications to have access to various parts of a device's functionality or software APIs, security breaches, such as an application being given access to a sensitive area of the device's operating system can cause damage to the actual system. One of the problems faced by the developer's in this case are having the resources to implement a more secure system for their application, such as a remote server to store information, simply at times due to the cost of the hardware involved. Time and experience is also essential, and it will be difficult for the developer to completely test the application for any gaps or exploits in its security, as many different problems may crop up in the live environment which they may not be able to test for using an emulator, or in a controlled environment using an actual physical device. An inexperienced developer may also be developing an application that will store user information on the device, but due to their lack of familiarity with developing for the mobile platform, or even general security protocols, may leave gaps in their code, or application (for example – an SQL injection), which could be exploited by hackers.

USER INTERFACE

Usability is a key and often overlooked factor in the design of User Interfaces (UI). The design of the UI in mobile devices is especially important and, while it may not be immediately the first thing the user is interested in when looking for an app to carry out a task, it subconsciously affects their enjoyment of using the app and determines whether or not they will return to using it. Because of this, usability is an area which is only really now being highlighted when it comes to mobile application development. Mobile devices have very small screens, especially in comparison to desktop computer, therefore the typical information and content displayed on a desktop's monitor would be unable to fit onto a mobile devices screen and still be presentable, as the screen would be cluttered and some things would not format or fit on the screen at all. This calls for a radical re-imagining of the way we display this information on the smaller display as well as choosing what content would not be suitable for this smaller platform and how to substitute for this. This is one of the biggest difficulties facing the developer when it comes to user interface design.

We can see from Figure 4, while they may be from different applications and have different functionality; they both have a distinct UI. On the

Figure 4. UI design examples



left the UI is cluttered, with too many buttons on the screen, which may cause the user to incorrectly touch the wrong button due to the screens small size and how close together the icons are. It also gives no indication as to what the icons mean, which could be confusing to a first time user. On the right is a very simple, clearly defined set of buttons, with indications of what they are and attractive icons. There is also very little chance of the user hitting the incorrect button, as they are very large and clearly defined. These two apps when contrasted show how much of a difference good UI design can go in improving the usability of an application. For the user, this will affect their interaction with the product greatly, as this will provide their dialogue with the application, and as a developer, it will be a challenge to provide both an attractive and functional UI that carries out exactly what the user wants the application to do without any distractions or over-complication of the process.

Of course, designing a successful application for mobile devices goes beyond simply what you can fit onto the screen, the user's interaction with the device a whole has to be considered. The design principles commonly associated with traditional interaction with software has to be re-thought. We usually take for granted the common drag-and-drop, cursor, scroll bars and text entry mechanisms of a traditional desktop for granted and generally associate software with naturally being interacted with in these ways. This is a problem faced with when designing for a mobile device, as this functionality generally doesn't port over well to a mobile device. As a developer, designing methods around which a user will interact with their application will take considerable planning and ingenuity. Although users may be used to direct manipulation to interact with software, developers shouldn't be constrained by this, but should look for alternative methods, such as personalisation of the applications to provide the user with information they need, or prompts based on that users preferences or meta-information (Onias-Kukkonen, 2003). Developing like this is ultimately challenging, but a method that is increasingly popping up and becoming almost characteristic of mobile development in these early stages.

FUTURE RESEARCH DIRECTIONS

Mobile devices are becoming more prevalent as our main source of information and applications are becoming the gateway to which we can access it. This trend will continue into the near future, which will drive developers to follow and create numerous applications for these mobile devices. With every application created the developers will face the barrage of problems, issues and choices. These should not be ignored but taken into consideration when developing an application for a mobile device thus improving the overall quality, security and accessibility of the application. The mobile device industry is very much fragmented, which leaves developers making many difficult decisions in order to maximize their applications success and availability. The large corporations that control the market places do not agree on many standards and protocols, which is making the divide within the mobile device ecosystem larger. Security is becoming the main concern for the large mobile device providers as more and more people are using mobile devices this is becoming a new market for hackers and cyber criminals. Many developers leave their applications vulnerable and many do not patch their application, which puts the users of the application at great risk of malicious intent. One of the main aspects that is cropping up amongst the mobile apps community is personalisation and user centeredness. This can be achieved by analysing the user's behaviour and preferences and identifying areas where the applications can prompt the user with a suggestion based on these, or by automatically providing information that they may want and minimizing text entry (e.g. drop down boxes). This kind of approach, as mentioned earlier, is strongly becoming characteristic of mobile applications, and this can help bypass several of the limitations of the device and its UI if implemented correctly in the development stages. This design approach should be strongly emphasised in future application development as it will lead to more functional, user friendly applications. As apps are implicitly characteristic of compactness, ease of use, speed and functionality, it is important that the final product is able to deliver on these ideals if it is to be successful. It is important to the user that the application carries out exactly what is asked of it, with the minimum of fuss, and even just one annoying bug that impairs the app's functionality, even slightly, could turn the user off the product completely, seeking an application which does it better. This is why testing is crucial and any bugs which may visibly affect the running of the core functionality should be top-priority and stamped out immediately. By prioritizing bugs or potential problems by risk, based on what the developer imagines could go wrong with the application in the field, the developer can work towards trying to ensure that many of the main problems which may affect the running of the application, or the users enjoyment of the application, are avoided by writing test cases for these risks and testing for them, or tightening up the coding around these areas using error-handling code and other techniques. By carrying out a risk analysis during testing, the functionality of the final product can come out very stable, and as such the core functionality will be much less bug-prone, as the core functionality usually crops up as a high-priority risk, for which test cases can be built around and any problems resolved, as to ensure the user a silky smooth experience with the application. This kind of testing should be especially implemented into development for mobile applications, as the many problems associated with developing for the platform could be eased if the developer can draw up some, even if rough, test cases for possible failures on the field, and prevent these from happening on the field.

Larger companies can spend money on specific, specialised testing done by humans, such as hiring a company to do their testing for them, improving the quality of final product and its appeal to users. On the flipside indie developers can spend more time on their product, as they are not usually restrained by time restrictions and can more openly discuss with their user base about what their feedback is on the application during its development and use this to improve their applications. With independently developed apps, the developer can usually develop a more unique or inspired product which may find a niche in the market, as the fact that they are developing an app may have come from inspiration of what is needed in the market, rather than a more business focused, productivity driven application. In either situation though, it is important to keep the scope of the project very focused, and not get distracted by complicating the design with any unnecessary functions or UI elements. This simplification of the plans means that the core functionality will receive more attention during development and should lead to a more polished and streamlined product. Unnecessary features will also need more time to develop and test and may cause compatibility issues.

Simplification of the project plans will also mean that the hardware requirements of the application will be lessened, and in the case of a resource costly application, will enable it to run on older or less capable devices. This streamlining should also apply to the User Interface. As the UI will provide the dialogue between the applications functions and the user's needs, it is important that the user can tell the application exactly what to do, and the app being able to produce results quickly and conveniently. This means that good UI design is tantamount to a successful app. For an app to thrive, the developer needs to know his target audience. In UI terms this boils down to what devices people are using and how best to make use of what hardware there is on these devices. Of the input methods concerned with mobile devices, the main ones that have to be considered when deciding how the user will interact with the application are the buttons. Hard buttons and soft buttons, as well as any shortcuts these may enable should function similarly to that device's native functions. This design prospect will require more time and effort in research but will ultimately increase usability of the app and increase user's enjoyment of using it (Dumaresq & Villenueve, 2010).

Uploading an application to an app market is a big step in the development cycle of an application, and it is up to the developer to take charge of how the application will be advertised after it is uploaded. As well as constraints placed on what can be uploaded to an app store, based on the platform, the developer will have to include screenshots and information about the application itself to give users a snapshot of what the app does and how it functions. This is a very important stage in selling the product to its intended audience, as there may be thousands of other apps like it, it is important that the functionality is concisely described and the actual software itself stand up to the description. Research into the applications intended audience, as well as preferred platform will ensure that the project id more focused and will be more likely to reach its intended audience. There are tools available such as Google analytics, which have APIs that can be imbedded into the app to measure various things such as where the app is popular, downloads, average usage time etc. can be used for research for future apps, descriptions for apps based on how they are used, giving information that can give an idea of what can be implemented into patched, or even run in a controlled test environment with a selection of users to see what functionality is being used, and what can be trimmed out of the final product.

CONCLUSION

Through critical analysis of the difficulties faced when developing mobile applications, it becomes apparent that much of the problem is our limited understanding of how to translate our traditional techniques of software development over to this platform and how this reciprocates through many different facets of application design. Some recommendations can be drawn from this conclusion. Firstly security seems to be constantly overlooked, especially in lower budget applications. Security should not be optional but mandatory and an application should be analyzed or open to a series of standard tests by some sort of body to ensure it is acceptable for the common user. With the mobile market being so vast and having so much potential it is very easily getting out of hand, all mobile applications should adhere to some sort of general standards, which again would be monitored by a committee and would create a stepping-stone towards unifying many of the fragmentation present today. User Interface design principles should be available as a resource for developers but should not be enforced, enforcing can hamper creativity and the originality of the application developer. Hopefully in the near future we will see many of these problems that developers are experiencing when creating applications for mobile devices being addressed. Allowing more robust applications to be created quicker, to a higher standard and making them more accessible therefore improving the experience for both the developers and end users.

REFERENCES

B'far, R. (2005). *Mobile computing principles: designing and developing mobile applications*. Cambridge, UK: Cambridge University Press.

Burns, J. (2008) Developing Secure Mobile Applications For Android. iSec Partners.

Dumaresq, T., & Villenueve, M. (2010). *Test Strategies for Smartphones and Mobile Devices*. Mississauga, Canada: Macadamian Technologies.

Dwivedi, H., Clark, C., & Thiel, D. (2010). *Mobile Application Security*. New York: McGraw-Hill.

Fling, B. (2009). *Mobile Design and Development*. Cambridge, MA: O'Reilly Publishers.

Hakoama, M., & Hakoyama, S. (2011). The impact of cell phone use on social networking and development among college students. *The American Association of Behavioral and Social Sciences Journal*, *15*(1), 58–66.

Holzer, A., & Ondrus, J. (2009). Trends in Mobile Application Development. In C. Hesselman, C. Giannelli, O. Akan, P. Bellavista, J. Cao, F. Dressler, D. Ferrari, et al. (Eds.), Mobile Wireless Middleware, Operating Systems, and Applications – Workshops, (Vol. 12, pp. 55-64). Springer Berlin Heidelberg.

Jacobs, M. (2011). *Living on the Edge of Mobile Development.* Retrieved from http://java.sys-con.com/node/1719019

Kumar Jha, A. (2007). *A Risk Catalog for Mobile Applications*. BookSurge Pub.

Lowe, A. (2011). *Hacking on the rise – all around.* Retrieved from http://hexus.net/business/news/general-business/32399-hacking-rise-around/

Mark, D., Nutting, J., & LaMarche, J. (2011). *Beginning iPhone 4 Development Exploring the iOS SDK.* Apress Pub.

Meier, R. (2010). *Professional Android 2 Application Development.* Wiley Publishing Inc.

Onias Kukkonen, H. (2003). *Developing successful mobile applications*. Stanford University.

Pettey, C., & Goasduff, L. (2011). *Gartner Says Sales of Mobile Devices in Second Quarter of 2011 Grew 16.5 Percent Year-on-Year; Smartphone Sales Grew 74 Percent*. Retrieved from http://www.gartner.com/it/page.jsp?id=1764714

Volk, J. (2010). *Make Money Online With Affiliate Marketing*. Retrieved from www.jonathanvolk.com/4Xel2Sf9mj/jvolkaffiliateguide.pdf

Wasserman, A. (2010). *Software Engineering Issues for Mobile Application Development*. ACM Digital Library.

KEY TERMS AND DEFINITIONS

HTML5: HTML5 is a core technology markup language of the Internet used for structuring and presenting content for the World Wide Web. It is the fifth revision of the HTML standard (created in 1990) and, as of December 2012, is a candidate recommendation of the World Wide Web Consortium (W3C).

Human Computer Interaction (HCI): Human–computer interaction (HCI) involves the study, planning, design and uses of the interaction between people (users) and computers. It is often regarded as the intersection of computer science, behavioral sciences, design, media studies, and several other fields of study.

JavaScript: JavaScript (is a dynamic computer programming language. It is most commonly used as part of web browsers, whose implementations allow client-side scripts to interact with the user, control the browser, communicate asynchronously, and alter the document content that is displayed.

Location-Based Service (LBS): A Location-Based Service (LBS) is an information or entertainment service, accessible with mobile devices through the mobile network and utilizing the ability to make use of the geographical position of the mobile device.

Mobile Application Development: The process by which application software is developed for low-power handheld devices, such as personal digital assistants, enterprise digital assistants or mobile phones.

Protocol: An agreed-upon set of rules that facilitates the exchange information between two computers or devices. A protocol includes formatting rules that specify how data is packaged into messages. It also may include conventions like message acknowledgement or data compression to support reliable and/or high-performance network communication.

Standard Generalized Markup Language (SGML): An international standard in markup languages, a basis for HTML and a precursor to XML. SGML is both a language and an ISO standard for describing information embedded within a document.

Universal Resource Identifier (URI): The string (often starting with http) comprises a name or address that can be used to refer to a resource. It is a fundamental component of the World Wide Web.

Usability: This focuses on the creating of the system making sure it is useable. Using the user's experiences and the factors involved this helps to create a better view into the development of technology, and help provide and maintain a better experience for the user with future technologies.

W3C Consortium: The World Wide Web Consortium (W3C) is the main international standards organization for the World Wide Web (abbreviated WWW or W3).

Web Service: A Web Service is a software component that is described via WSDL and is capable of being accessed via standard network protocols such as but not limited to SOAP over HTTP. It has an interface described in a machine-processable format (specifically WSDL).