A Protocol for Evaluating Mobile Applications

Clare Martin, Derek Flood, Rachel Harrison
Oxford Brookes University, United Kingdom

ABSTRACT

The number of applications available for mobile phones is growing at a rate which makes it difficult for new application developers to establish the current state of the art before embarking on new product development. This chapter outlines a protocol for capturing a snapshot of the present state of applications in existence for a given field in terms of both usability and functionality. The proposed methodology is versatile in the sense that it can be implemented for any domain across all mobile platforms, which is illustrated here by its application to two dissimilar domains on three platforms. The chapter concludes with a critical evaluation of the process that was undertaken.

1 INTRODUCTION

Portable devices, such as mobile phones and music players, are now capable of running a wide variety of applications (or 'apps') which enable users to perform a broad range of tasks while they are away from traditional computing devices. This has contributed to the staggering growth of the mobile phone market in recent years. In October of 2010 it was estimated that by the end of that year there would be 5.3 billion mobile subscriptions worldwide, and that in developed countries there would be 116 subscriptions for every 100 inhabitants on average [1]. It was also estimated in 2010 that 8.2 billion applications would be installed on mobile devices worldwide. It is projected that by the year 2015 this value could increase to as much as $32 billion, [2] which is a significant increase in such a short space of time.

One reason for the recent growth in this market is that mobile applications can be developed cheaply, and in a relatively short time frame. This is partly due to the simplicity of the dominant mobile platforms, such as iOS from Apple and Android from Google, and also because the associated software development tools are freely available. Consequently, users are often faced with a broad choice of applications to help them complete a given task. A recent survey [3] identified four factors which influence users when choosing an application: function, price, opinions of others and usability. Developers should therefore consider all of these factors when designing a new application, but it can still be difficult to evaluate the current status of the market for any given domain, since each one is so densely populated.

Before a project can be started, some effort is needed to determine the groundwork necessary for requirements engineering and the context in which it takes place [4]. The context is particularly important for mobile applications development since it is essential to consider the socio-technical environment in which they will be developed and used. The groundwork can include an analysis of the present state of the art through a systematic survey of existing applications. This chapter describes a flexible, cost-effective protocol that can be used to perform such a study quickly and thus eliminating the vast amount of irrelevant applications in a domain.

The proposed evaluation methodology utilises expert evaluators rather than users. It comprises a series of steps that can be used to filter through a collection of applications by comparing functionality, efficiency and various other attributes affecting usability such as personalization, ergonomics, flexibility, security and error management. In addition, the process was designed to elicit functional requirements by generating a list of features offered by existing applications. The protocol is both platform and task independent which is demonstrated here through its application to two separate task domains on three platforms.

This chapter is organised as follows. Section 2 contains some background information about usability evaluation, with particular reference to mobile devices. The evaluation protocol is then introduced in Section 3, followed by a summary of the results of the two case studies in Section 4. A critical evaluation of the protocol is discussed in Section 5, which is followed by some suggestions for future research in Section 6. Section 7 then concludes this chapter.

2 BACKGROUND

Mobile devices are hand-held tools which typically have a graphical display with input via touch, stylus, miniature keyboard, or some combination of these methods. Examples include personal digital assistants (PDAs), smart phones, music players such as the iPod Touch and tablet computers such as the iPad and Kindle Fire. The study of the usability and design issues associated with such devices is still in its infancy, since they are very different from desktop computers, both in terms of interaction mechanisms and other attributes such as context, connectivity, small screen size, display resolution and limited processing capability [5]. The major platform providers, such as Apple and Google, have produced extensive guidelines [6, 7] for developers of mobile applications, and there are also independent guidelines that focus specifically on improving the user experience [8]. However, previous research suggests that current techniques for the evaluation of such technology lack structure and that there is a need for a systematic approach using a combination of methods, since no single technique gives answers to all design questions [9].

Standard techniques for usability evaluation can be grouped into three categories [10]: controlled settings involving users, natural settings involving users, and settings which do not involve users, but instead use consultants and researchers. There are a number of different ways to test user interfaces in each of these three categories, including observation, interviews, questionnaires, logging, heuristics and walkthroughs. The protocol described in this chapter falls into the third category: it is designed to be conducted by experts rather than users, and the standard procedures that it utilizes are keystroke level modelling and heuristic evaluation.

The keystroke level model (KLM) [11] was invented as a way of allowing individuals or companies to give quantitative predictions of the time expert users would take to complete certain tasks, thus reducing the need for expensive user studies. It is closely associated with the GOMS (Goals, Operators, Methods and Selection) model [12], . The GOMS model operates by first selecting a goal, or task, that is to be achieved, such as inserting a formula into a spreadsheet. There are often several methods for achieving such a goal, and each one is broken down into steps. The operators involved in each step, such as mouse clicks and key presses, are then identified, and selection rules are used to decide which method to choose. For example, when summing a small group of cells in a spreadsheet it might be quicker to select the group using the mouse, but for a larger group it might be quicker to type in the formula. KLM is closely related to GOMS in that it can be used to quantitatively compare different ways of achieving the same goal using different methods, applications or devices.

KLM was first developed by Card et al. [11] through empirical studies to determine a set of approximate times that each of the most commonly used operators would be expected to take. The original KLM model contained four such operators:

- K :- keystroke or button press;
- P :- pointing to a target on the display with a mouse;
- H :- homing the hand(s) on the keyboard or other device;
- D :- drawing (manually) straight line segments;

Average times were also suggested for other components of a typical task execution, such as mental preparation and system response time.

Once a goal has been broken down into steps involving operators via GOMS, KLM can be used to give a consistent prediction of the time it will take to carry out using different methods or with different applications. The operators used in the original model were all associated with desktop computing, but a number of studies have proposed an extension of the original model to include the kind of operators that have been introduced by mobile computing. For example, [13] includes predictions of the time taken to carry out tasks using a stylus on a Palm PDA, and [14, 15] provide predictions for mobile phones. However, the rapid evolution and variety of the interaction mechanisms on these devices means that this is still an ongoing area of research.

Heuristic evaluation [16] is another way of measuring the usability of an interface, whereby a small group of usability experts examine different aspects of the interface in relation to standard design principles, or heuristics. The number of experts involved in such an evaluation can be as small as three since it has been shown [16] that three to five experts are sufficient to uncover most usability problems. The evaluations take place independently, and the results are then analysed to detect the range and severity of the various usability problems. It is a cost-effective evaluation method in comparison with some other popular techniques, but it does have limitations since the participants are not real users and it does not fully capture the context of use. The set of heuristics to use when evaluating an interface can depend on the domain of use; the list originally developed by Nielsen [16] provides a good starting point, and can be supplemented by extra heuristics developed from design guidelines, or from knowledge of the domain. Bertini et al performed a study of the usability issues associated with mobile applications [17] and produced a new set of mobile usability heuristics, shown in Table I:

Table I: Mobile usability heuristics (from [17])

Heuristic	Description
A	Visibility of system status and losability/findability of the device
B	Match between system and the real world
C	Consistency and mapping
D	Good ergonomics and minimalist design
E	Ease of input, screen readability and glancability
F	Flexibility, efficiency of use and personalization
G	Aesthetic, privacy and social conventions
H	Realistic error management

These heuristics have been systematically developed and empirically validated to take into account some of the differences between mobile devices and traditional computing devices. For example, the small screen size means that only crucial information should be displayed, which is encompassed by heuristic D. This contrasts with typical desktop design, which can be *lazy* in the sense that everything can be put on the screen, leaving the user to determine what features are important. For the purposes of the two case

studies described here, each mobile heuristic was broken down into sub-divisions to be assessed by each evaluator, some of which were specific to each domain; more details are given in Section 4.2.

3 THE EVALUATION PROTOCOL

The methodology used here was devised as a way of allowing experts to filter the large number of mobile applications in each domain to obtain a set that was small enough to be evaluated in greater depth through an adapted form of KLM and then heuristic evaluation. The purpose of the method is not just to compare the usability of each product but also to elicit functional requirements. The protocol was influenced by the usability standard for medical devices ISO/IEC 62366, which describes a usability engineering process with nine stages, two of which involve identifying key functionality. First, the frequently used functions, or *red routes* [18], are identified, followed the primary operating functions, which includes all the frequently used functions but also functions that may affect the safety of the device. These functions are identified in the third step of the protocol described below and the functions provide the goals upon which the KLM and heuristic evaluation are subsequently based. An intermediate fourth step has also been introduced here, to capture additional functional requirements by examining all the extra functionality offered by the various applications. The steps of the protocol are as follows:

1. *Identify all potentially relevant applications*. This step consists of searching the applications related to a particular keyword. Current online stores, such as the App Store℠ from Apple, the Android Market from Google and the App World from BlackBerry, facilitate this task. It is recommended that this search be performed through the web-based interfaces, as some on-device store interfaces do not present all mobile applications available.

2. *Remove light or old versions of each application.* The trial versions (those that offer a subset of the functionality offered by the corresponding full application or access to the full application for a limited period of time) should be removed as the full version will be evaluated.

3. *Identify the main functional requirements and exclude all applications that do not offer this functionality*. This functionality includes both frequently used functions and also other functions that are essential but infrequently used, such as language and unit settings. Only the applications that meet all the main requirements are carried forward to subsequent steps of the protocol.

4. *Identify all secondary requirements.* This step consists of identifying any functionality offered by each application that is not part of the main list of functional requirements identified in the previous step. The purpose of this secondary list is to gather a comprehensive list of possible functionality for a developer to analyse during the requirements elicitation process for any new application.

5. *Construct tasks to test the main functional requirements using each of the methods below*:
   - *Keystroke level modelling* (KLM) is used to provide a measure of efficiency for each application. Since the new interaction methods provided by mobile devices have not all been incorporated into KLM, it is not possible to predict efficiency in terms of time, however it is possible to use the number of interactions as an indirect measure of the efficiency of each application.
   - *Heuristic evaluation* is used to give a qualitative analysis of the possible usability problems. The heuristics shown in Table I were developed specifically for mobile applications and are therefore appropriate for this analysis.

4 PROTOCOL VALIDATION

The above protocol was validated through its application to three mobile device platforms: iOS from Apple, Android from Google and Blackberry OS from Blackberry, and in two application domains: diabetes management and spreadsheet applications.

4.1 APPLICATION DOMAINS

The first application domain for the validation was healthcare. Applications were studied that allow people with diabetes to manage their condition by logging daily information, such as blood glucose level, carbohydrate intake and insulin dose. The second application domain was business. Spreadsheet applications which can allow users to complete a wide variety of tasks from financial planning to statistical analysis were chosen from this domain. The following sections outline these two domains.

4.1.1 Healthcare: Diabetes Management

Type 1 diabetes occurs when the insulin producing cells of the pancreas are destroyed leaving the body unable to control its blood glucose levels. People with type 1 diabetes have to take insulin regularly to try to stop their glucose levels from becoming too high, but if they take too much insulin their glucose levels may also drop too low, causing a number of symptoms including dizziness and palpitations.

The vast majority of patients with type 1 diabetes administer their insulin through multiple daily injections, and the remaining proportion use insulin pumps. Most people are offered a structured education programme, such as DAFNE [19], to help them self manage their condition. This teaches them how to calculate the amount of insulin to administer at each meal according to the current blood glucose level, number of carbohydrates consumed and various other factors such as time of day, exercise and illness. The daily glucose levels are often stored in a hand-written diary which is shared with the healthcare team at regular intervals. It is surprisingly difficult for patients to keep their blood glucose levels within the target range, and yet failure to do so can lead to serious complications which are a huge burden on the health service.

Most insulin pumps come with dose calculators to help patients determine how much insulin to administer, but people on multiple daily injections do not usually have this support, and tend to do the calculations themselves. This trend is beginning to change, with the advent of glucose monitors such as the Accu-Chek Expert [20], manufactured by Roche, and the Insulinx monitor from Abbott [21] which do have dose calculators, but the prohibitive cost has meant that these have not yet become widely used. Consequently, a growing number of mobile phone applications have been developed both to log healthcare data and to offer help with calculations.

This domain was selected because it represents the broader category of productivity applications which have a narrow focus and a limited range of primary functionality, which are common features of mobile applications. Other examples of such applications include applications for note taking, calorie counting and scheduling lists.

4.1.2 Business: Spreadsheets

Spreadsheets are ubiquitous software tools used for a variety of tasks from financial planning to statistical analysis. The mobile nature of business is increasing the need for users to access spreadsheets while on the move. Therefore mobile spreadsheet applications are becoming more important and the requirements of users are expanding to include more advanced functionality such as specialist functions and features.

A recent survey [22] has shown that most users only use mobile spreadsheet applications for examining existing spreadsheets, such as those they receive in an e-mail, or for altering existing spreadsheets to examine the impact that changes will have on the models contained within these spreadsheets.

The small screen size associated with mobile devices is one of the biggest issues associated with mobile spreadsheet applications. However, survey participants identified a number of contexts in which the only possibility was to view the spreadsheet on a mobile device. Examples of such contexts provided by the participants included discussing changes in assumptions, commuting, and viewing e-mail on a mobile device.

The spreadsheet domain was selected to represent general productivity applications where the specific task will vary greatly from user to user. Other examples of such applications include those for text creation, photo editing and music creation.

4.2 RESULTS

The following sections outline the results obtained at each step for both domains.

Step 1: *Identify all potentially relevant applications*

The first step of the protocol requires selection of appropriate key words for each domain which are used to search the app stores of each of the chosen platforms. In the case of diabetes management the keyword *diabetes* was used while the word *spreadsheets* was used to search for spreadsheet applications.

	iOS	Android	Blackberry
Diabetes	231	168	28
Spreadsheets	105	179	58

Table II: Number of apps returned by App store search

Table II shows the number of applications that were returned for each of the key words searched. The table shows that a large number of applications were available for both domains on the iOS and Android platforms but only a relatively small number for the Blackberry.

Step 2: *Remove light or old versions of each application*

After compiling the results obtained from each of the searches, any *light* or old versions of other applications included within the search results were removed because these versions often contain less functionality than the full or newer version. Table III shows that there are only a small number of such versions available in each domain.

	iOS	Android	Blackberry
Diabetes Management	9	6	1
Spreadsheets	16	14	10

Table III: Number of light or old versions

Step 3: *Identify the main functional requirements and exclude all applications that do not offer this functionality.*

The next step of the protocol involves the identification of applications that offer the main functional requirements that are required for a given domain. For the diabetes management, these requirements were as follows:

1. Set Measurement Units;
2. Log Blood Glucose Level;
3. Log Carbohydrate Intake;
4. Log Insulin dose;
5. Display Data graphically;
6. Export data via email or similar.

The reason for choosing the first requirement is that different units are used for measuring insulin in the United States and Europe and many applications only offer the units that are used in the country where the application was developed. The next three requirements are for logging blood glucose level, carbohydrate intake and insulin dose as recommended by medical professionals. Viewing this data graphically can help these patients identify trends which can help to avoid health problems in the future. The final functional requirement was to export the data so that users can back-up their entries in case their mobile device is lost or damaged. This is a crucial part of the health management process.

For the spreadsheet application the main functionality was the ability to view and alter existing Microsoft© Excel spreadsheets. A recent survey [22] has shown that users typically require mobile spreadsheet applications to either view an existing spreadsheet, received by email or to alter an existing spreadsheet model to see the effect of changes to the input data.

	iOS	Android	Blackberry
Diabetes Management	8	6	1
Spreadsheets	11	7	1

Table IV: Number of applications offering main functional requirements

The search for applications with the required functionality was conducted by examining the descriptions of each application on the online app store. Table IV shows that only a small number of the applications offered this functionality. The remaining applications were designed for a wide variety of purposes. In the case of diabetes management these applications included general health information, cookbooks with recipes specific for diabetes sufferers and Body Mass Index (BMI) calculators.

For the spreadsheet search a number of training books and video applications were returned along with those for specific purposes such as calculating tips. In addition to this a number of applications were returned which exported data to spreadsheets, for example *Contacts Export* allows users to export some or all of their phone contacts to a spreadsheet. These results highlight the inefficiency of existing search algorithms on mobile app stores. The algorithms used are not freely available and as such it is difficult to determine why they are so ineffective.

Step 4: *Identify all secondary requirements*

The fourth step of the protocol consisted of identifying all of the secondary functionality of the evaluated applications. This is carried out in order to allow developers to analyse the full range of functionality offered by competing applications and to determine opportunities for additional features, which can help to differentiate a new application from existing ones.

Within the diabetes management domain a wide range of secondary requirements were identified. It is beyond the scope of this chapter to include a full list of these requirements but the interested reader is referred to [23] for further details. Some of the most common secondary requirements are listed below:

- *Log physical activities and other medications.* These factors can impact the appropriate dose of insulin that needs to be taken and therefore a number of applications allow users to store this type of information.
- *Allow personal settings:* Most applications allowed users to input personal information such as email address, target glucose level and weight.
- *Carbohydrate database:* Some of the applications contained a carbohydrate database which allows users to look up particular food items to determine the amount of carbohydrates contained. This is important in the diabetes domain as users are required to calculate the amount of insulin required from the amount of carbohydrates consumed.
- *Insulin dose calculator*: Some applications feature the ability to calculate an appropriate dose of insulin to inject based on the amount of carbohydrates consumed and the blood glucose level.

A range of secondary functionality for spreadsheet applications was also identified, most of which is already available in desktop spreadsheet applications. A summary of the key secondary functionality is presented here.

- *Freeze panes:* Freeze panes allow a user to keep a number of rows and/or columns locked on screen while they move through the rest of the spreadsheet. This feature is particularly useful when a user needs to keep headings available on-screen while searching through a large spreadsheet.
- *Sorting:* A number of applications allow users to sort tables of data on a given column. The small screen size associated with mobile applications makes it difficult to view all of the required data at one time, therefore making it necessary for users to re-order the information to see what they are most interested in.
- *Go to cell:* When looking at large spreadsheets navigation can be a time consuming task. A number of applications allow users to type a cell address and the application will move the user to this cell, saving a large amount of time.
- *Formatting:* A number of spreadsheet applications allow users to format cells on the spreadsheet. They allow users to set the foreground and background colour of the cell, change the font and size of the text contained within the cell as well as making it bold and italic.

One notable omission in the functionality of applications in either domain is the ability to filter data. When examining a large quantity of data, such as that associated with patient records, users often need to limit the data in order to focus on a particular subset. The limited screen size of mobile devices can make it particularly important to limit datasets so that users can examine them more easily.

Step 5: *Construct tasks to test the main functional requirements using each of the methods below:*

The final step in the protocol was designed to examine the usability of the existing applications both in terms of efficiency and through adherence to heuristics listed in Table I.

Each of the applications from Step 3 was downloaded from the relevant app store in order to conduct the efficiency testing using a form of keystroke level modelling. This was carried out by a single evaluator since it is purely a quantitative measure. The four most efficient applications from each domain on each platform were then subjected to a further heuristic evaluation by between three and five evaluators. The

reason for excluding the remaining applications was partly to do with resources and partly because of duplication of the same problems among the various applications. By restricting attention to a maximum of four applications in each domain we reduced the number of applications to 24 and the number of evaluations to 120. The following sections outline the main findings for each of the two stages of the usability evaluation. For a full discussion of the results the interested reader is referred to [23].

Tasks

Each of the evaluations was based upon a series of tasks that were devised to test the primary functionality associated with each domain in Step 3. For diabetes management, the tasks were as follows:

1. Change the insulin units to "mmol/L";
2. Add a new entry with the following values:
    a. Blood glucose level of 6.7 mmol/L;
    b. Carbohydrate intake of 50 grams or 5 portions;
    c. Insulin dose of 5.5 units.
3. Display a visual representation of the data;
4. Export logged data via email.

As spreadsheets can be used for a wide variety of tasks, a simple series of tasks was created which examined the most common aspects of spreadsheet usage. First, a simple spreadsheet was created containing five numbers in the first column together with a sum formula to total the five numbers. This allowed the examination of the following task.

1. Entering a single digit value in a cell;
2. Entering a formula in a cell.

KLM evaluation

The original method of KLM described in Section 2 associates approximate times with each of the common operators used in desktop computing applications. New interaction methods are developed frequently for mobile applications however, and some of them, such as the swipe action, have not yet been incorporated into the traditional KLM model. Therefore it was not possible to predict the time it would take to complete the tasks according to the operators used but instead the number of interactions to complete each task was used as an indirect measure of its efficiency.

The results for the diabetes applications are detailed in [23], and showed that the efficiency of mobile applications varies greatly. For example on the iOS platform it was found that it took 3 interactions to set the units to 'mmol/L' on one application but with another application this same action took 10 interactions. The is because the RapidCalc application requires users to exit the application and enter the device settings to change the units while other applications contain their own settings tab.

Similar results were found in the spreadsheet application where it took between 4 and 13 keystrokes to enter the sum formula. This variation is because the most efficient applications allow users to select 'sum' from a menu so that the system can enter the appropriate formula. In contrast the least efficient application requires the user to manually enter entire formulae, including cell references.

One similarity between all of the spreadsheet applications was in the choice of input methods. This was typically the keyboard, but in the diabetes domain there was a wide range of input methods used for entering the insulin, glucose and carbohydrates. Direct input through the keyboard, sliders, and pickers were all used for entering the same information in different applications.

Heuristic Evaluation

The heuristic evaluation was conducted by giving each evaluator a mobile device with the relevant applications installed, and an evaluation sheet upon which to record results. The evaluators were asked to carry out the selected tasks independently at least once. The evaluation sheet contained a section for each mobile heuristic, as defined in Table I, which was then broken down into domain specific sub-divisions. For example the subdivisions of Heuristic E for the diabetes domain were as follows:

- E1 It is easy to input the numbers
- E2 It is easy to see what the information on each screen means
- E3 You can easily navigate around the app
- E4 The screens have a 'back' button
- E5 The user can get crucial information 'at a glance'

Details of the other sub-divisions are given in [24]. Each attribute was given a ranking using Nielsen's Severity Ranking Scale [25], which was corroborated by evidence and justification for that ranking.

The heuristic evaluation exposed a different set of problems from the KLM, because the mobile heuristics that guided it are not all related to efficiency. It is beyond the scope of this paper to give a full qualitative summary of the evaluation, which is presented in [25], but the most commonly breached heuristics for the diabetes applications were A, C, D, G and H (Table I). This led us to the following associated guidelines:

- The battery status and time should be visible while using the application.
- The network status should be visible while sending data.
- Provide options for backing up and restoring data.
- Do not include unnecessary options or irrelevant information.
- Do not overload a screen with too many elements.
- All data transmission should be encrypted.
- Do not allow unrecoverable errors.

In the spreadsheet domain a number of heuristics were also breached. The most commonly breached heuristics were B, E, F, G and H. One of the most common problems uncovered during the evaluation was the data input method. In most applications, when a user tries to input data the system presents them with the alphabetic keyboard rather than a numerical one, despite the most common form of input in a spreadsheet being numerical values. As with the diabetes domain a number of guidelines were developed:

- It should be possible to encrypt data and files.
- The input method should allow quick entry of numerical data.
- Symbols should be distinct and not easily confused.
- Users should be able to undo previous actions.
- The menu structure should be easily discovered.

5 CRITICAL EVALUATION OF THE PROTOCOL

This section contains a summary of some of the advantages and disadvantages of this protocol, together with an outline of some of the difficulties that were encountered when using it.

This evaluation protocol was devised in order to gain a broad overview of the status and trends of all of the mobile applications currently available for a given task domain. The results of the previous section have shown that the method gives a useful way to quickly reduce the search for applications offering key functionality in each domain, and to give a systematic evaluation of those that remain. Even though the style of keystroke level modelling used here was not sophisticated, it provided a very quick way to compare the efficiency of the applications before embarking on a more in-depth analysis of the better performing applications using heuristics. Some of the differences in efficiency highlighted by the KLM were caused by the devices themselves, and others were due to the design choices of each implementation. For example the hard keyboard of the Blackberry proved to be surprisingly less efficient than the touchscreens of the other two devices. Examples of implementation decisions that affected efficiency included the choice of data entry method and default settings. The heuristic evaluation uncovered a different set of problems from the KLM since most of the heuristics used were not related to efficiency. Instead, some fundamental problems were exposed concerning security, error recovery, minimalist design, visibility of system status and backup/restore options. The analysis of secondary functional requirements generated many recommendations in each domain, which could then be further distilled using additional requirements analysis techniques.

5.1 ADVANTAGES

The protocol has a number of strengths. For example it is highly adaptable in the sense that it is platform independent and need not even be restricted to a single mobile platform. The first step relies on the existence of an app store, but such distribution mechanisms are now becoming commonplace for desktop and web applications as well as for mobile platforms since the inception of the Mac App Store, Chrome Web Store and forthcoming Windows 8 App Store. The protocol also offers flexibility in terms of application domain as the two contrasting case studies described here have shown.

As well as being flexible, the protocol establishes the state of the art within a specific task domain in a cost effective manner. It focuses on applications that provide a core set of functionality of interest to developers thus eliminating a large number of applications returned by a preliminary search.

The protocol is economical since it can be applied by a small number of people, indeed the majority of the work can be conducted by a single individual. The only exception to this is the heuristic evaluation, for which multiple usability experts are required. But since this is the final step of the protocol, only a small number of applications are evaluated and therefore the time commitment required by the experts is relatively small.

The protocol outlined above has proved to be cost effective not only in terms of money but also in terms of time, at least in the latter stages, because so many applications are eliminated right at the start (Step 3). Consequently developers do not waste time looking at the high volume of irrelevant applications that are returned by the initial search of the app stores. The results presented in Section 4 have shown that the identification of key functionality is a particularly useful mechanism for excluding a large number of applications from subsequent evaluations.

The time needed to learn how to apply the protocol is low. During the evaluation presented above two postgraduate students were introduced to the protocol and quickly became efficient with its application. By the time they applied it to the second platform they required little supervision or guidance and were comfortable using the protocol on their own.

The protocol produces a comprehensive set of the secondary functionality provided by existing applications. So developers of new applications can see which functionality is most common, and which therefore should be considered for inclusion within the new application, and which functionality is missing, and which might thus provide an opportunity for the new application to differentiate itself from existing competition. Since the protocol gathers functionality from multiple platforms, it may uncover features that do not yet exist on any applications on the platform for which the new application is to be developed.

The application of the protocol described in Section 4 showed that the use of a combination of two different analytical usability techniques generated an extensive list of pitfalls to avoid. The KLM evaluation exposed some fundamental efficiency issues whereas the heuristic evaluation showed that there were problems concerning security and error recovery. The two techniques therefore complemented each other in generating a wide range of issues associated with each task domain. Although further issues can be uncovered through the use of user studies, such studies can be expensive and time consuming to perform. The approaches adopted here can be performed in-house using limited resources, which is an important consideration for small software development companies which are typical within the mobile application domain.

5.2 DISADVANTAGES

The protocol has some weaknesses. For example, some of the steps are time consuming, and may be tedious to perform. Also, it is not always possible to determine the full functionality of each application from its description on the online store, but this is an important step in reducing the number of applications to purchase in order to avoid incurring high costs. It can be difficult to identify all of the relevant keywords to use in the initial search, yet a poor choice can return a high number of unconnected applications, or mean that the search is not exhaustive. There are also problems with measuring the number of keystrokes used to carry out a task, since it may depend on the user's dexterity when using certain kinds of interaction objects such as pickers and sliders. Since the heuristic evaluation is qualitative rather than quantitative some results are necessarily subjective, according to the personal taste of the evaluator.

The problem with the keyword search might be avoided by conducting multiple searches to ensure that the broadest range of applications is returned. However, in this case the developer will encounter the same application multiple times. One way to circumvent this problem would be to automate the search procedure. Through an automated searching script, multiple searches could be done together, presenting the user with a complete set of unique applications returned by any search term.

Although the protocol is designed to be efficient, the volume of results returned by the searching algorithms may mean it takes time to evaluate all of the applications returned. It would be possible to use app store ratings to further refine the results to eliminate some of the lower quality applications but this may result in key functionality being missed.

Mobile applications are released on a frequent basis. This means that the results obtained will be time sensitive. Through automated search tools, a system could poll for new applications on a daily basis during the development lifecycle. This way when new applications are released the developers can evaluate these upon release, which can be done quite quickly for individual apps.

. It also tends to highlight the problems with applications and ignore their positive features and so information may be lost. Such functionality might be elicited by involving users in the evaluation for example, but this was avoided in order to reduce the time and cost associated with the process.

5.3 THREATS TO VALIDITY

This protocol is not intended as a complete method for eliciting requirements or designing an interface since it has some known and deliberate limitations. Most obviously, it ignores any kind of user evaluation, including the star rating of the app store. It also excludes price. As a method of gathering requirements it is restricted in the sense that it only examines the functionality of existing applications. The results could be used to form the basis of subsequent requirements gathering by informing the design of questionnaires and structured interviews. With regard to interface design, it can be useful in showing which interaction objects work best for each task, and which designs to avoid, but it is no substitute for a traditional design involving iterative prototyping.

Another limitation of the experiment described here is that the results were restricted to applications running on the iOS, Android and Blackberry smart phone platforms. These were chosen because they have the highest market share in the UK [9], and jointly occupy 75.5% of the total market. It was also restricted to native applications, since such applications can take advantage of device-specific features which are particularly interesting to developers.

There was also a difficulty when searching the Android App store from the mobile device. When a search was conducted on the mobile device, a lower number of applications were returned than when the same search was performed using the app store website through a desktop computer. This may have happened because the app store limited the results to only those applications that would function with the appropriate version of Android running on the handset.

6 FUTURE RESEARCH DIRECTIONS

This protocol is likely to evolve and change in the future: it is a prototype which will need refining and improvement through further empirical studies. For example, the version of KLM used here was quick and effective but unlike traditional KLM it did not use predictive timings. Instead it used a simple count for each operator, so more research is needed to determine reliable timings for the commonly used interaction objects.

The heuristic evaluation was also relatively fast to perform, but it is only one of many other evaluation techniques that could have been chosen including analytics and cognitive walkthrough. A recent usability study of smart phone applications [26] found that a hybrid of heuristic evaluation and cognitive walkthrough, dubbed *heuristic walkthrough* was also an effective technique. Therefore a future iteration of this methodology could include additional or modified steps involving alternative techniques.

The potential for future research in the broad domain of usability evaluation of pervasive applications is enormous. A number of suggestions are given in [5] including study into the best ways to design presentation methods, menu structures, database facilities, data entry methods and connectivity issues. In addition, some of the problems of precision using pickers and sliders noted in Section 4 suggest that fundamental experiments are still needed to determine the optimal granularity of such objects in the same way that much research took place in the early days of GUIs to decide how many buttons to include on a mouse.

7 CONCLUSIONS

Requirements gathering is an important phase of any software development project. When developing for mobile devices however it may be too costly and time consuming to evaluate all existing applications within a given application domain. To address this issue this work presents a systematic evaluation protocol for determining the state of the art in a given application domain. This protocol has been applied to two domains: healthcare and business, to give a detailed snapshot of the status and trends in a subset of each domain. It has been found to be both cost effective and efficient for evaluating the large volume of applications within these two domains. The protocol has been applied to three of the most popular mobile platforms in the UK and it was found that on each platform the protocol enabled an efficient examination of the two application domains.

Although further usability testing should be done to determine all of the usability issues with existing apps, the protocol highlighted a number of issues that should be avoided when developing new applications within these domains and helped to produce some guidelines for the design of mobile applications.

Acknowledgments
The authors would like to thank Oxford Brookes University for generous support towards this research.

REFERENCES:

1. ITU, *The world in 2010 ICT Facts and Figures*. 2010, ITU.
2. Junniper Research, *A world of Apps*. 2010.
3. Derek Flood, Rachel Harrison, and David Duce, *Using Mobile Apps: Investigating the usability of mobile apps from the users perspective.* International Journal of Mobile HCI [Submitted], 2011.
4. A. Finkelstein, *Requirements Engineering: an overview*, in *2nd Asia-Pacific Software Engineering Conference (APSEC'93)*. 1993.
5. Dongsong Zhang and Boonlit Adipat, *Challenges, Methodologies, and Issues in the Usability Testing of Mobile Applications.* International Journal of Human-Computer Interaction, 2005. **18**(3): p. 293 - 308.
6. Apple, *iOS Human Interface Guidelines*. 2011, Apple Inc.
7. Android Developers. *User Interface Guidelines*. 2011 [cited 20th October 2011]; Available from: http://developer.android.com/guide/practices/ui_guidelines/index.html.
8. Nielsen Norman Group, *Report Usability of Mobile Websites & Applications:*.
9. *Online Marketing Trends*. [cited 28th October 2011]; Available from: http://www.onlinemarketing-trends.com/2011/04/smartphone-marketshare-2011-italyus-aus.html.
10. Y. Rogers, H. Sharp, and J. Preece, *Interaction Design 3rd Edition*. 2011: John Wiley & Sons Ltd.
11. Stuart K. Card and Thomas P. Moran, *The Keystroke-Level Model for User Performance Time with Interactive Systems.* Communications of the ACM, 1980. **23**(7).
12. Stuart Card, Thomas P. Moran, and Allen Newell, *The Psychology of Human Computer Interaction*. 1983: Lawrence Erlbaum Associates.
13. L. Lou and B. E. John. *Predicting Task Execution Time on Handheld Devices Using Keystroke Level Modelling*. in *CHI 2005 Extended Abstracts on Human-Factors in Computing Systems, ACM* 2005.
14. Trenton Schulz, *Using the Keystroke-Level Model to Evaluate Mobile Phones*, in *IRIS 31*. 2008.
15. Paul Holleis, et al., *Keystroke-Level Model for Advaned Mobile Phone Interaction*, in *CHI 2007*. 2007.
16. Jakob Nielsen and Rolf Molich, *Heuristic Evaluation of User Interfaces*, in *CHI 1990*. 1990.

17. E. Bertini, et al., *Appropriating Heuristic Evaluation for Mobile Computing* International Journal of Mobile Human Computer Interaction (IJMHCI), 2009. **1**(1): p. 20-41.
18. *Red Route Usability: The key user journeys with your web site*. [cited 28th October 2011]; Available from: http://userfocus.co.uk/articles/redroutes.html.
19. DAFNE. [cited 25th Febraury 2011]; Available from: http://www.dafne.uk.com/.
20. ACCU-CHEK. [cited 21st February 2011]; Available from: http://www.accu-chek.co.uk/gb/products/metersystems/avivaexpert.html.
21. *Abbott Insulinx*. [cited 28th October 2011]; Available from: http://www.abbottdiabetescare.co.uk/your-products/freestyle-insulinx.
22. Derek Flood, Rachel Harrison, and Kevin McDaid, *Spreadsheets on the Move: An evaluation of mobile spreadsheets*, in *The European Spreadsheet Risk Interest Group (EuSpRIG) Annual Conference*. 2011.
23. Eva Garcia, et al., *Systematic analysis of mobile diabetes management applications on different platforms*, in *USAB 2011 - Information Quality in eHealth*. 2011.
24. Clare Martin, et al., *A Systematic Evaluation of Mobile Applications for diabetes management using heuristics.* [Submitted].
25. J. Nielsen. [cited 28th October 2011]; Available from: http://www.useit.com/papers/heuristic/heuristic_evaluation.html.
26. A. Cousin, *The contribution of heuristic walkthrough compared with user study to a smartphone usability evaluation.*