DOCTOR OF PHILOSOPHY

SUM

an exploration of Shared User Models and Interface Adaptations to Improve Accessibility of Mobile Touchscreen Interactions

Montague, Kyle

Award date:
2013

Awarding institution:
University of Dundee

DOCTOR OF PHILOSOPHY

SUM

An Exploration of Shared User Models and Interface Adaptations to Improve Accessibility of Mobile Touchscreen Interactions

Kyle Montague

2013

University of Dundee

SUM:

An Exploration of Shared User Models and Interface Adaptations to Improve Accessibility of Mobile Touchscreen Interactions

Kyle Montague

Doctor of Philosophy

University of Dundee

2013

Contents

List of Tables

List of Figures

Acknowledgements

It is with immense gratitude that I acknowledge the support and guidance of my supervisors Prof. Vicki L. Hanson, and Andy Cobley. For all of your help getting me through this thesis, and your kindness and care through health concerns. Vicki you have been so much more than a supervisor, I cannot find words to express my appreciation and admiration for you. It has been an absolute honour and privilege to work along side you. I know this would not have been possible without you. For that I am indebted.

This work would also not have been possible had it not been for the time, support and feedback of my study participants, your input has been and will always be greatly appreciated. I would also like to acknowledge Paciência Cristiano Canda and Rachel Montague for their software application development and expertise.

I share the credit of this work with my colleagues within the SiDE research group, and School of Computing. Marianne Dee from Dundee, to you I owe a special thanks. You have contributed more than you realise. Thank you for the user evaluations that ran smoothly, for the words of encouragement and side splitting laughter. BISCUITS! Thanks to Dr Lesley McIntyre for your kind words of inspiration and reassurance, your bottomless bag/drawer/cupboards of confectionery treats, and the chats with coffees and cakes (especially the ones we accidentally stole.) To Claire Jones for regularly reminding me that I'm not as great as I

sometimes like to believe, for occasionally telling me that I'm great when I didn't feel it. To Dr Ha Trinh for always being there and offering your precious time to help me with my problems, and letting me sneak your in-flight snacks and beers while you sleep. To Dr Hugo Nicolau I can't thank you enough for your involvement; you have been a tremendous support in the final stages of my PhD journey.

Special Acknowledgement to Dr John Richards, you always made time to lend a hand and offer advice. While I may not thank you for pointing out my movie star doppelganger, I can't thank you enough for your continued involvement throughout this period of my life. I would like to thank Ann Kenny and Anne Miller, you troublemakers have been a large part of Dundee life.

An enormous thanks to my family and friends for the continued support and patience throughout my PhD. Thanks to my sisters Jo-anne, Rachel and Sarah for the countless phone calls assessing my sanity and providing much needed distraction. A massive thank you to Hazel and Scott Montague for always supporting me throughout my education, for being fantastic role models and parents. **I could not have done this without you**.

Finally, I would like to acknowledge the support provided to this research by RCUK Digital Economy Research Hub EP/G066019/1 – SIDE: Social Inclusion through the Digital Economy.

Declaration by the candidate

I declare that I am the author of this thesis; that, unless otherwise stated, all references cited have been consulted by me; the work which this thesis records is mine; and that it has not been previously presented or accepted for a higher degree.

Kyle Montague

2013

Declaration by the supervisor

I declare that Kyle Montague has satisfied all the terms and conditions of the regulations made under Ordinances 12 and 39; and has completed the required 9 terms of research to qualify in submitting this thesis in application for the degree of Doctor of Philosophy.

Professor Vicki L. Hanson

2013

Abstract

Touchscreens are ever-present in technologies today. The large featureless sensors are rapidly replacing the physical keys and buttons on a wide array of digital technologies; the most common is the mobile device. Gaining popularity across all demographics and endorsed for superior interface flexibility of soft designs and rich gestural interactions, the touchscreen currently plays a pivotal role in digital technologies. However, just as the touchscreen has enabled many to engage with digital technologies, its barriers to access are excluding many others with visual and motor impairments. The contemporary techniques to address the accessibility issues fail to consider the variable nature of abilities between people, and the ever-changing characteristics of an individual's impairment. User models for personalisation are often constructed from stereotypical generalisations of the similarities of people with disabilities, neglecting to recognise the unique characteristics of the individuals themselves. Existing strategies for measuring abilities and performance require users to complete exhaustive training exercises that are disruptive from the intended interactions, and result in the creation of descriptions of a user's performance for that particular instance.

This research aimed to develop novel techniques to support the continuous measurement of individual user's needs and abilities through natural touchscreen device interactions. The goal was to create detailed interaction models for individual users, in order to understand the short and long-term variances of their abilities and characteristics, resulting in the development of interface adaptations that better support interaction needs of people with visual and motor impairments.

This thesis describes the development and evaluation of the Shared User Model (SUM) Framework, developed to help improve the access and usability of touchscreen devices by people with visual and motor-impairments. The framework is intended to be embedded by application developers to create interaction models autonomously and provide suitable interface adaptations to better support individual users with visual and motor-impairments. The SUM Framework captures the user's natural application interactions in the form of low-level touch and device movements, and then starts to model their individual interaction characteristics to refine the gesture recognisers and tailor these interactions to the needs of the user.

The outcomes of this research stem from three foundational user studies. The first study represented the initial requirements gathering and problem scoping stage of this research, helping to better define the barriers and challenges to touchscreen technologies. Findings from this study formed the basis for the SUM Framework, targeting the interaction challenges faced by people with visual and motor-impairments when using mobile touchscreens. The second study was devised to evaluate the principle low-level interaction modelling approach of the SUM framework, and gain further insight of the variances between users within stereotypical groupings. While this study was pivotal to the development of the SUM framework methodologies, the entire evaluation took place within a controlled laboratory environment. The research concluded with a much longer four week in-situ evaluation to address the limitations of the short timescale laboratory study, and investigate the potential of SUM as a long-term solution for modelling users with highly volatile abilities.

This research presents the potential benefits of the SUM Framework to create more accessible touchscreen interactions, supported by rigorous user evaluations from the laboratory and in the wild. Finally, the thesis outlines a number of directions and areas for future research expanding on the concepts developed within this work.

Associated publications

Montague, K., 2010. Accessible indoor navigation. In *ASSETS '10: Proceedings of the 12th international ACM SIGACCESS conference on computers and accessibility*. ACM, pp. 305-306.

Montague, K., Hanson, V.L. and Cobley, A., 2011. Adaptive interfaces: a little learning is a dangerous thing... *Universal Access in Human-Computer Interaction. Design for All and eInclusion*, Part I , Volume Part I, pp. 391–399.

Montague, K., Hanson, V.L. and Cobley, A., 2012. Designing for individuals: Using the SUM Framework. In *Designing Interactive Systems '12*. DIS '12.

Montague, K., 2012. Interactions speak louder than words: shared user models and adaptive interfaces. In *Adjunct proceedings of the 25th annual ACM symposium on User interface software and technology* (UIST Adjunct Proceedings '12). ACM, New York, NY, USA, pp. 39-42.

Montague, K., Hanson, V.L. and Cobley, A., 2012. Designing for individuals: usable touchscreen interaction through shared user models. In *ASSETS '12: Proceedings of the 14th international ACM SIGACCESS conference on computers and accessibility*. pp. 151-158.

Chapter 1. Introduction

This work has been motivated by two fundamental challenges. Firstly the need for better user models, more representative of individuals' current needs and abilities. User models need to be capable of responding to the continuously fluctuating abilities of individuals, this is particularly important for individuals with disabilities. Secondly, there is a need for better touchscreen accessibility. Regardless of the mainstream popularity of touchscreen devices they still pose challenges and barriers to access for many disabled and able-bodied users alike. The primary objective of this research is to develop techniques that allow user models to capture and respond to individual abilities seamlessly, in turn improving the accessibility and usability of touchscreen technologies.

1.1 The Need for Better User Models

User models are primarily used to personalise or tailor application content and interface designs to better meet the needs of the end user. Typically a user model would consist of information relating to the preferences, knowledge or abilities of the end user that was otherwise unavailable during the design stages of the application. Therefore, systems can be designed to adapt and respond to contextual information to improve the interaction experience for the user. However, this approach to designing relies on the premise that the information within the user model accurately represents the current interaction propensities and requirements of the user.

1.1.1 Representing Real-World Users

User Diversity. Within industry it is uncommon for companies to involve disabled users in the design and development of new products and technologies. The adopted strategies are to select individuals who are representative of much wider populations to take part in design investigations and evaluations. While user centred design is encouraged more and more in industry, the levels of involvement and diversity of users is inadequate for the purpose of designing technologies that work for disabled users. Interfaces are, at best, designed to meet the needs of generic abstractions of a disability, neglecting full consideration of the range and daily fluctuation of symptoms and characteristics. The task of creating an interface that accommodates for all these factors quickly becomes more complex and costly.

Mobile Conditions. In addition to the human behaviours, there are external factors that will influence interactions (e.g. lighting conditions, travel conditions); for mobile technologies the number and variety of environments makes it impossible to predict and design for all eventualities. Thus there is an increasing need for interfaces that are able to adapt and respond on a more dynamic and individual basis. This can only be achieved through systems that respond rapidly to changes in conditions.

Continuous Calibration. User models are commonly used to tailor application content and presentation at use time, applying contexts that were not present during the design phase of applications, in order to provide a better user experience. There are many different types of data a system might want to model; an individual's preference for one interface layout over another, the current task workflow, domain

specific knowledge in e-learning environments, measurements of the user's typing performance with a QWERTY keyboard, and so on. The type of model data inevitably defines the scope and style of customisation possible. Likewise there are a number of methods to capture the user model data; explicitly asking the user or providing configuration features, including code to recognise domain specific workflow sequences, requesting that the user complete a calibration task, or implicitly capturing performance measurements from application activities. The method for capturing user model data has tremendous implications for the user model's ability to respond promptly and in line with the needs of the user. Offering user configuration options or initial calibration phases are among the most common technique for capturing user data. Traditionally occurring within the first launch of a device or application, these enable the user to explicitly define settings and provide the system with a baseline measurement which is often never to be revisited. While this initial configuration or calibration might have been sufficient for desktop interactions of able-bodied users, this one time setup is insufficient for representing users with variable abilities such as people with motor impairments. It also neglects to consider the impact of the user's current situation, which can be particularly variable when considering mobile technologies. However, the current approaches to user modelling would require the user to recalibrate or reconfigure the device in each specific situation.

1.1.2 Why Touchscreens?

Although the motivation for improvements to user models is independent of any single technology, this research focuses on mobile touchscreen devices due to their mainstream appeal and increasingly ubiquitous nature. More than 500 million

touchscreen units shipped in 2012, with mobile devices accounting for 34% of all units, and numbers are predicted to reach 660 million by 2015 (Cellular-News, 2008). Smartphone devices represented 60.1% of the smart connected devices shipped in 2012, with tablet devices, portable PCs and desktop PCs representing 10.7%, 16.8% and 12.4% of the market share respectively. The worldwide market share of smartphones increased by 53.1% from 2011 to 2012 (IDC, 2013). In 2011 the UK alone was reported to have 25.4 million smartphone users, a penetration rate of 51.3%, which rose to 64% in 2012 (comscore, 2012a; 2013). The trends suggest that users are moving away from traditional mouse and keyboard PC interfaces for accessing content and towards mobile touchscreen interactions (comscore, 2012b).

Touchscreens have great appeal due to their ability to support new forms of human interaction, including the interpretation of rich gestural inputs and the rendering of novel user interfaces. However, the technology creates new challenges and barriers for users with limited levels of vision and motor control due to its lack of tactile cues. Furthermore, it relies on the user's ability to accurately and consistently perform the rich gestural inputs in alignment with the predefined parameters of the gesture recogniser; for some users this is not always possible. Although there are alternative devices and specially augmented hardware solutions to improve the accessibility of touchscreen interaction (such as screen overlays), they result in further exclusion from mainstream technologies, and threaten to stigmatise those who use them.

1.2 Approach

This dissertation explores techniques to produce user models that are representative of the diverse interaction abilities of individuals, continuously responsive to the short-term variances affecting user performance, and present minimal interruption to real-world interactions, with the goal of addressing the existing challenges with touchscreen technologies by individuals with visual and motor impairments.

1.2.1 Sharing Data

To mitigate the risk of user models becoming out of date and requiring user interruption to recalibrate, the proposed methods aim to reuse and share interaction data between applications and users where possible. Rather than defining user models that are specific to each application and reliant on domain knowledge to function, this approach leverages the interaction behaviours that are common amongst applications by decomposing their interfaces and interactions into the low-level components and gestures such as buttons and taps, allowing an application-independent user model to be created. Similarly, the component measurements can be leveraged to permit the sharing of data between users, independent of their stereotypical groups. The major challenge of sharing data between applications and users is insuring that the data is useful and is going to help define the interaction abilities of the current user. It is not enough to simply select all examples of button taps and train the user model. By including users' data where the interaction behaviours are significantly different from the current user's abilities, the resulting user models could produce further recognition problems and failed interactions. Therefore, techniques to select the appropriate training data are required.

1.2.2 Contextual Modelling

In order to support the sharing of user data between applications and users, a novel method for selecting training data that matched the current interaction context was proposed. The approach uses a small sampling window to measure the interaction behaviours for each session. Features are extracted from these measurements and used to identify previous sessions with similar interaction behaviours, thus selecting relevant data to train the user models on. This method is independent of user and application and therefore allows user models to be trained from other users' data. Furthermore, it increases the availability of interaction data for training and enabling the development of user models specific to individual situations.

1.2.3 The SUM Framework

The Shared User Modelling (SUM) framework captures measurements of an individual's interaction performance through real-world application interactions, allowing continuous measurements of users' needs and abilities. For example, SUM framework measures the duration of an individual's onscreen taps to identify the range of durations that define an intentional tap gesture for that individual, and adapts the parameters of the tap gesture recognisers to meet the individual's interaction behaviours. Interaction measurements are domain independent, thus SUM models can easily be shared between applications. The SUM Framework also contains the necessary methods to apply user models and tailor application interactions, removing the need for designers to have any knowledge of user modelling or interface adaptation. SUM allows disabled users to interact with the same technologies as able-bodied users, providing touchscreen interactions that are

more sympathetic to individual abilities using off the shelf mainstream technologies, thus reducing the risk of exclusion or stigmatisation of disabled users.

At an abstract level, SUM is a technique combining domain independent user models and adaptive interfaces to personalise touchscreen interactions. SUMs are built through background processing of real-world application interactions, as opposed to subjecting users to semantically meaningless calibration exercises to elicit performance measurements. This dissertation demonstrates the application of SUM to improve the access of touchscreen devices for people with low levels of vision and motor ability. While there is a long history of interface adaptations for disabled users, such efforts have focussed largely on adaptations for a specific disability or device. SUM parameterises user interactions to define individual models of input behaviours rather than relying on stereotypical user group characterisations and impairment assumptions. Although this work focuses on touchscreen interactions by users with visual and motor impairments, it aims to demonstrate the wider application of this technique within other technologies involving users with different interaction challenges. However, in its current state SUM is presently limited to physical abilities.

The contribution of the present work is the proposal and evaluation of the novel Shared User Modelling (SUM) approach as well as the design and development of the SUM framework that implements novel user modelling and interface adaptation methods. The SUM framework provides adaptations based an individual's current and fluctuating needs. SUM enables the measurement of user abilities and interaction characteristics without the need for separate calibration exercises or explicit user

configuration settings. Shared user models can be built using interaction data across multiple applications using input data captured from other users. The research reported here focuses on a mobile touchscreen device, reflecting the growing prevalence of such devices and the user interaction challenges they pose for many.

1.2.4 Research Approach

This research has adopted an iterative user centred design approach to define and refine the SUM framework. Incremental versions of the SUM framework were evaluated through user observations, laboratory studies and concluded with a four week in-situ study involving users with visual and motor impairments. Interaction logs were captured using the SUM framework; these were combined with pre- and post-evaluation discussions, gathering additional user information and interaction feedback. Refinements were made based on both quantitative device data and qualitative user data, ensuring design changes were based on supportive data evidence. The structure of this dissertation outlines the user evaluations and resulting changes to the SUM framework.

1.3 Thesis Aims

The main objective of this thesis work is to explore and develop techniques to improve the accuracy of user models to increase access and usability of mobile touchscreen interactions by people with visual and motor impairments. Applying a user centred and iterative design process, the research resulted in the creation of the SUM framework. The SUM framework serves as a self-contained user modelling and interface adaptation tool, designed specifically to model real-world application interactions and tailor the interface to meet the current abilities and needs of

individual users. The SUM approach shares interaction data between users and applications to train new user models based on the contextual measurements of the current interaction abilities of users.

Thus, the thesis proposed in this dissertation is:

Sharing data between users and applications can produce models that usefully represent the dynamic needs and abilities of individuals.

1.3.1 Key Research Questions

To investigate the proposed thesis the following research questions were defined:

- What are the common touch interaction characteristics and individual variances of users with visual and motor impairments, and how can affordances for individual abilities be made to improve touchscreen interactions?
- Can user abilities be accurately captured and modelled through natural interactions within mobile touchscreen applications?
- How can user models respond to short-term changes and fluctuations of user abilities and needs, without the need for continuous calibration exercises?
- Can measurements of users' abilities be applied to improve the accessibility of touchscreen interfaces?

1.4 Contribution to Knowledge

The research presented in this dissertation provides a comprehensive review of the current accessibility state of mobile touchscreen technologies with respect to visual

and motor impairments, as well as a critical reflection on the application of current user modelling and adaptive interface techniques. Furthermore the methodologies and evaluations conducted within this work impart new insights into the benefits, and challenges faced, when conducting user studies outside of the controlled laboratory environment. The work also presents a tangible proof of the utility of the SUM Framework, a system developed to capture and monitor user interactions for modelling and adaptation purposes.

The contributions to the field of accessible human computer interaction (HCI) of this thesis are:

- The exploration of interaction monitoring and modelling techniques to support the creation of user models built from real-world application interactions, leveraging the otherwise discarded low-level touch behaviours within the gesture recognisers to develop a rich understanding of a user's abilities and interaction characteristics, thus removing the reliance for calibration activities to train and update user models.

- The development of a domain-independent structure for user models to support the sharing of user information between applications and touchscreen devices, and a software framework to utilise the user modelling and adaptation capabilities of the model structure.

- The rigorous evaluation of interaction modelling without the need for measurement elicitation tasks, on touchscreen devices by people with visual

and motor impairments, both within a controlled laboratory environment and a real-world setting through a four week in-situ user study. These studies provide a rich understanding of how users' interaction behaviours and abilities fluctuate both in short-term and long-term device usage.

- A detailed provisioning of procedures and tools to aid the transition from laboratory to in-situ user evaluations of mobile touchscreen devices.

- The proposal and evaluation of a novel approach using contextual measurements of user interactions to create user models specific to individual sessions. The contextual models allow the creation of user models from other users' data, therefore are independent of stereotypical disabilities.

1.5 Thesis Structure

Chapter Two describes prior work in the field and its relation to SUM. The chapter also aims to provide the reader with a snapshot of the current state of mobile touchscreen technologies, exposing the challenges and barriers to access. It highlights the contemporary methods to address the accessibility issues, from both an industrial and academic perspective. Finally, the chapter outlines and discusses the characteristics and abilities associated with the user population involved throughout this research, to give the reader an understanding of the breadth and variability of these characteristics and challenges to technology access, discussing the contemporary approaches to designing for these populations and drawing the reader's attention to the impact of poorly considered designs.

Chapter Three details a preliminary study carried out with older adults acting as further requirements gathering beyond the knowledge gained from the literature review. The preliminary study helped to scope the interaction challenges presented by touchscreen interactions, and refine the background user modelling process.

Chapter Four leads on from the background modelling techniques discussed within Chapter Three, and provides a complete technical overview of the devised SUM Framework for modelling users' interactions through natural interactions. This chapter goes on to detail the development of the domain-independent model structure supporting the sharing of user models between applications, and concludes by scrutinising the limitations and other considerations of such user modelling techniques.

Chapter Five discusses a laboratory-based study carried out with visual and motor impaired users, to evaluate the use of the SUM Framework to develop adaptive interfaces tailored to individuals (published in Montague, Hanson, & Cobley, 2012). It concludes with a critical reflection of the laboratory-based study, discussing the limitations of such studies and barriers to transitioning SUM user evaluations into the wild outlining the necessary provisions to support in-situ evaluations of SUM.

Chapter Six outlines the development changes made to the SUM framework, addressing the limitations identified from the laboratory user evaluation discussed in Chapter Five. The chapter details the new provisions made to the SUM Framework to support application in the wild.

Chapter Seven discusses the design and execution of an in-situ user study with visual and motor impaired users to explore the real-world behaviours and fluctuations of individuals' abilities.

Chapter Eight introduces a novel user modelling approach, leveraging the interaction behaviours from individual sessions to produce user models that are specific to the current interactions of the user. The chapter evaluates the proposed models using simulations from the user data from the in-situ study of Chapter Seven.

Chapter Nine concludes the thesis and discusses the conducted research with relation to the research objectives and hypotheses outlined above. It presents a critical reflection of the findings and limitations; and details provisions for future research.

Chapter 2. Related Work

This chapter aims to express the importance and need for accurate user modelling methods in the on-going struggle for accessible technologies, with the focus of this thesis being on touchscreen interactions. Concentrating on key works within the field of accessible digital technologies and user modelling, this chapter provides a review of relevant research. The chapter begins by discussing the larger philosophical approach of this thesis, presenting the existing challenges within this area of HCI. Next the chapter presents a critical review of current user modelling strategies and systems. The chapter then outlines on-going work within the field of touchscreen accessibility. Finally the chapter discusses each of these three areas in relation to the work within this thesis.

2.1 Ability-Based Design

The concept of ability-based design has only recently been proposed by Wobbrock et al. (2011) however, its core principles have been alluded to by other movements such as Harper's (2007) design-for-one. Harper (2007) examined the feasibility of *design-for-all* as a real-world solution to creating accessible technologies, stressing that the universal approach contradicts itself. The *design-for-all* approach argues that we should design with everyone and every situation in mind, but we know that human abilities and interaction situations are too diverse and broad to deal with in this way (Vanderheiden, 2000). In contrast, the principle underlying of *design-for-one* and ability-based design is simple: technologies should be designed in alignment with what the user can actually do. With ability-based design, the disability group/medical diagnosis of a person does not define his/her actual interface needs.

People are individuals, each with unique needs and abilities. Our interfaces, therefore, should reflect this.

Seven Principles of Ability-Based Design

STANCE	**1. Ability.**	Designers will focus on ability not *dis*-ability, striving to leverage all that users *can* do.	*Required*
	2. Accountability.	Designers will respond to poor performance by changing systems, not users, leaving users as they are.	*Required*
INTERFACE	**3. Adaptation.**	Interfaces may be self-adaptive or user-adaptable to provide the best possible match to users' abilities.	*Recommended*
	4. Transparency.	Interfaces may give users awareness of adaptations and the means to inspect, override, discard, revert, store, retrieve, preview, and test those adaptations.	*Recommended*
SYSTEM	**5. Performance.**	Systems may regard users' performance, and may monitor, measure, model, or predict that performance.	*Recommended*
	6. Context.	Systems may proactively sense context and anticipate its effects on users' abilities.	*Recommended*
	7. Commodity.	Systems may comprise low-cost, inexpensive, readily available commodity hardware and software.	*Encouraged*

Figure 2.1 Seven Principles of Ability-Based Design extracted from (Wobbrock et al., 2011).

Wobbrock et al. (2011) describe the basic tenets of ability-based design in relation to other design approaches. They outline seven basic principles of ability-based design, shown in Figure 2.1. These principles outline the basic position of the approach, and suggest methods of interface and system design. This approach removes the barriers that define a user as being able-bodied or disabled, and instead invites designers to consider individuals' abilities, more specifically: *What can he/she do?*

2.1.1 Challenges in Ability-Based Design

Wobbrock et al. (2011) defined the following ability-based design challenges that closely relate to the research objectives of this thesis work. Firstly, the success of ability-based design relies on the system's ability to accurately detect an individual's abilities. There is a need for these abilities to be periodically measured with low cost

or disruption to the user's intended workflow. In relation to this, another challenge is to consider the user's current situation. It is important for applications to understand the wider context beyond the device. Successful ability-based design systems need to factor in the environmental factors that influence an individual's abilities. However, to truly achieve contextually-aware systems we must first develop methods to measure performance abilities away from the controlled laboratory setting, and traditional semantically meaningless calibration tasks. This presents what could be considered the greatest challenge, inferring an individual's intention within free-form tasks. Finally, once the aforementioned challenges have been addressed, it is crucial that the systems understand how best to behave in response to these measurements. These challenges constitute the rational and motivation for the research within this dissertation.

2.1.2 Measuring Abilities

Regardless of the task or application, arguably the most important stage of ability-based design is capturing accurate measurements of user abilities. The research presented within this section demonstrates techniques and strategies applicable to the accurate measurement of users.

Trewin and Pain (1997) presented a technique whereby an individual's typing behaviour and performance could be monitored in order to identify any keyboard difficulties such as long key pressing and bounce errors. Trewin extended this work and proposed novel filtering techniques to mitigate these types of errors, through techniques known as the *Dynamic Keyboard* (Trewin, 2002). This program provided optimal keyboard configurations for an individual's varying needs in roughly real

time. Targeted at individuals with motor impairments, the system was designed to cope with highly variable user abilities, requiring regular assessments of typing performance. The *Dynamic Keyboard* captured measurements by monitoring keyboard behaviour during users' natural interactions with the computer. This technique enabled the system to periodically assess the user's abilities without the need for any user interruption. Building on the original *Dynamic Keyboard* work Trewin (Trewin, 2004) demonstrated the adaptation capabilities of the system within a real world context. Using these measurements the system was then able to make modifications to the keyboard configurations helping to reduce typing errors such as *key repeats* from pressing delays, and *bounce errors* from unintentionally tapping keys multiple times.

Keates and Trewin (2005) recognised a similar need to support users with mouse clicking, reporting the most common types of errors as *slipping* when clicking, and *unintentional* or accidental clicks. Trewin et al. (2006) proposed a novel solution, *Steady Clicks,* which was able to significantly reduce these errors using filtering techniques to ignore mouse movements during clicks and accidental target selections. Hurst et al. (2008a) later developed methods to automatically capture and measure individual's mouse performance, and assess if there was a need for adaptation. Using Fitts' law-style pointing and clicking tasks the system collected movement and click behaviours and extracted features to classify participants as having motor problems, or no motor problems with a 92.7% accuracy. Furthermore they were able to predict with 94.4% accuracy whether *Steady Click* adaptations would be of benefit to the individual.

The *SUPPLE++* system used a similar Fitts' law-style calibration task approach to elicit the abilities of individuals, requiring the user to move a computer pointer to select various onscreen targets (Gajos, Weld, & Wobbrock, 2010; Gajos, Wobbrock, & Weld, 2008). These models of user performance were then used to automatically create interfaces that provided the most optimal interactions for the user as shown in Figure 2.2. SUPPLE++ provided adaptations such as scaling the size of controls and substituting checkbox controls for lists or buttons, all such changes being based on the performance abilities of the individual user. Gajos et al. (2008) tested the interface adaptation system with motor-impaired users producing significantly fewer errors and shorter completion times.

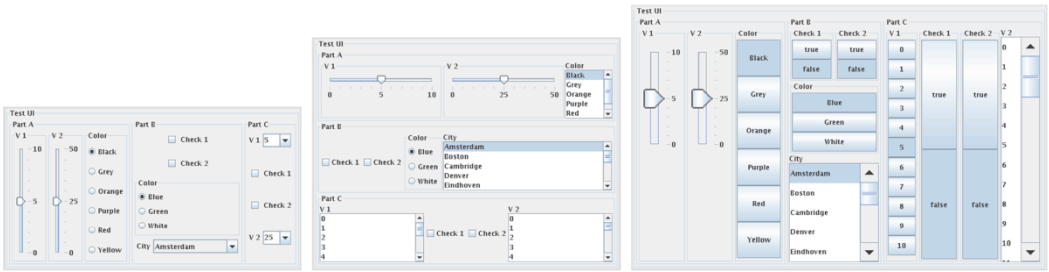

Figure 2.2 Baseline dialog interface and two interface versions automatically generated by SUPPLE++ extracted from (Gajos et al., 2008).

The major limitation of the method of determining adaptations used by Gajos et al. (Gajos, Wobbrock, & Weld, 2007), and one often used by others (Trewin S., 2004), is the need for users to complete a calibration task to inform the system of their current abilities. The very nature of some impairments is their highly variable behaviours. Thus, the individuals' abilities often have large variations. Using the proposed calibration technique to leverage user abilities may result in the need for users to undergo the task before each system use, and ultimately, fails to capture user needs that can change even within a session. Similarly abilities can be impacted by

the situation of use (Sears & Young, 2002), meaning that the application of this strategy in mobile technologies could inevitably result in users completing the calibration tasks with each use due to the impact of environmental factors alone.

2.1.3 Sensing and Adjusting to Current Situations

Sometimes the impact of the user's current situation can be overlooked as a factor when designing technologies. People do not pick up their desktop computer tower, keyboard, mouse and monitor then take it on a train ride. However, it is entirely plausible to take your laptop computer, tablet device or mobile smartphone into this situation. Moving away from the comfort of the living room or office exposes users to potentially harsh extremes of lighting, stability, ambient noise and distractions. Early efforts to capture environmental factors impacting on user interactions, such as light levels, resulted in the need for users to be heavily equipped with numerous sensors monitoring movements and locations using accelerometers, gyroscopes, cameras, light sensors and GPS chips (Roto et al., 2004). However, in recent years many of these sensors have become embedded in mainstream mobile technologies allowing the previously body worn sensors to be replaced with a single smartphone.

In relation to this widespread availability of portable situation sensing devices there has been an increase in the development of context aware systems. A number of efforts have all investigated the effects of situational impairments (Sears, Lin, Jacko, & Xiao, 2013) on user interactions.

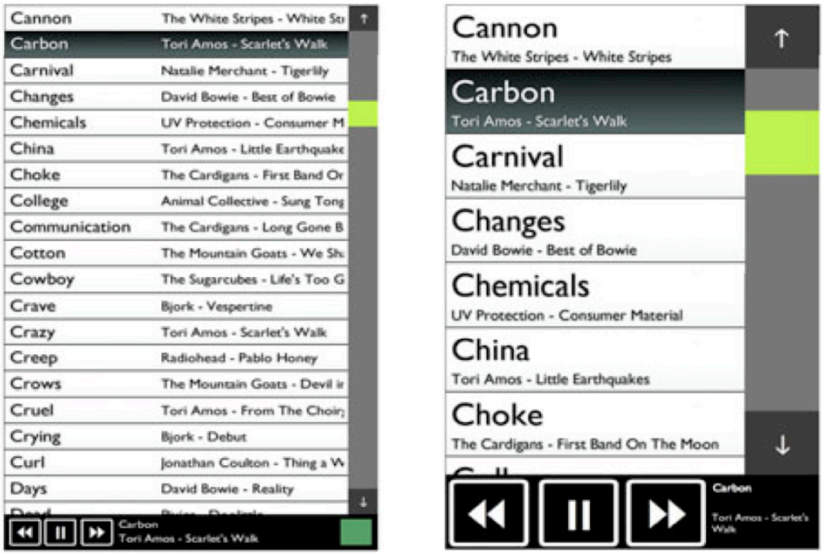

Figure 2.3 Walking User Interface for music application, extracted from (Kane, Wobbrock, & Smith, 2008).

Walking User Interfaces, shown in Figure 2.3, was the name given to mobile interfaces that adapted their form in relation to the users' movement (Kane et al., 2008). Using the device's built-in accelerometer sensor, the application interface increased the size of text and widgets as device movements increased. While producing larger widgets simplified the target selection task, it also caused fewer items to be visible at once on the screen. This forced users to perform a greater number of scrolling operations, resulting in longer task times when using the adaptive layout.

More recently, Nicolau and Jorge (2012a) investigated the effects of grip posture and movement on mobile touchscreen text entry. They used the built-in accelerometer to measure device movement and stability during repeated text entry tasks within single handed portrait orientation, two handed portrait orientation and two handed landscape conditions. Interestingly the authors reported that while two handed text entry increased the input rates, it provided no additional device stability or

improvement to accuracy of text-entry. Nicolau and Jorge (2012a) suggest that future techniques should focus on dealing with poor aiming.

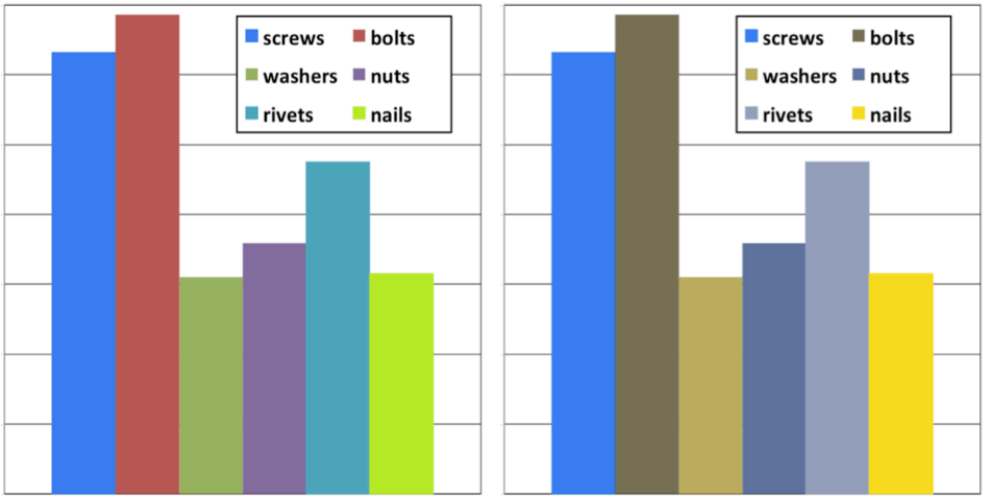

Figure 2.4 Original bar graph (left) and recoloured bar graph based on individual's colour vision abilities, generated using SSMRecolor (right) extracted from (Flatla & Gutwin, 2012).

While the aforementioned systems investigated mobile technologies and the situational impact of movement on interactions, other types of adaptations are possible. For example, Flatla and Gutwin (2011) investigated the effects of situation on colour differentiation with desktop interactions. The situation-specific models were constructed using a short calibration task that accounted for environmental factors such as lighting as well as the individual's own colour vision deficiencies. They later presented SSMRecolor, where the system used the situation-specific models to recolour interfaces tailored to an individual's colour vision abilities and current situation of use (Flatla and Gutwin., 2012). Figure 2.4 shows an example of bar chart recolouring. Using the situation specific models, participants were significantly more accurate at identifying differences between colours when compared against other colour correction methods. This work demonstrated the

nontrivial impact of environmental factors on the perception of visual interfaces, specifically colour discrimination.

2.1.4 Measuring in the Real World

Laboratory-based evaluations allow researchers to control for external factors that can influence participant interaction performance. Typically, these studies tailor situations to remove distraction and interruption thus ensuring a user's attention on the task and relative precision in interaction accuracy. While highly controlled laboratory experiments provide clean measurements with minimal errors, Chapuis et al. (2007) have demonstrated that interaction behaviours captured within natural settings differ from those captured within the laboratory. Additionally, laboratory-based evaluations impose time restrictions on user studies. Characteristically lasting no more than an hour at a time, they restrict the potential for capturing the performance changes that naturally occur throughout daily usage. During the *Dynamic Keyboard* evaluations, Trewin (2004) asked participants to provide typing samples at various points throughout the day to begin to understand these changes.

Hurst et al. (2008b) conducted "in the wild" user evaluations to investigate the pointing performance of individuals with motor impairments in natural usage conditions. The initial phase of the evaluation required participants to complete baseline calibrations using the IDA (Koester, LoPresti, & Simpson, 2005) software suite, based on Fitts' Law clicking tasks. Beyond this initial phase, participants were free to login to the system and play games, or use other applications such as word processing. Using application interaction models, the authors were able to infer user intent from the mouse input, allowing measurements of overlapping button clicks,

slips, accidental clicks, direction changes and excess distance travelled similar to the type of measurements possible within the controlled laboratory setting (Hurst, Hudson, Mankoff, & Trewin, 2008a). Hurst et al. (2008b) reported that participant performance was highly variable both between and within sessions, further supporting Trewin's early findings that individuals' performance can fluctuate due to medication, progression of a disease, or as a symptom of impairment (Trewin et al., 2006). Hurst et al. (2008b) argue that user evaluations with less control and constraints can help to reduce the risk of fatigue and stress by allowing participants to dictate their own break and interaction schedules.

More recently, Gajos et al. (2012) also explored real world user evaluations to develop techniques for collecting accurate measurements of pointing performance using unobtrusive methods, proposing that deliberate mouse pointing interactions could be distinguished from the "noisy" unintentional ones by extracting trajectories, speed, acceleration and jerk features of the mouse movements. Using online calibration tasks combined with natural data collection through a web browser plugin, the authors were able to develop filters and techniques to identify mouse interactions that occurred during periods of distraction allowing the collection of laboratory-quality data for mouse pointer measurements as used within the earlier SUPPLE (Gajos & Weld, 2004) evaluations.

2.1.5 Towards Accessible Interfaces

The use of suitable interactions is fundamental to the success of ability-based systems. Research described so far in this chapter has all explored interface and interaction techniques that maximise user accessibility or performance. However, not

all interaction techniques are beneficial to everyone; most interaction adaptations are defined for particular groups of people or disabilities. Although these interfaces are not optimal for everyone, they offer valuable insights into the correlation of interface adaptations and user abilities. Attention is now turned to the contrast between subject dependent studies (Guerreiro, Nicolau, Jorge, & Gonçalves, 2010a), which investigate interfaces targeted to improve access for a specific user population, and subject independent studies (Findlater & Wobbrock, 2012; Goel, Findlater, & Wobbrock, 2012), that expore interfaces as a response to parameterised measurements and abilities – typically adaptive systems.

Guerreiro et al. (2010a) examined the interaction challenges faced by tetraplegic people when using mobile touchscreen devices. Recognising that existing touch key models (Parhi et al., 2006, S. Lee & Zhai, 2009 and Y. S. Park et al., 2008) did not work for this user population, the authors conducted in-depth laboratory evaluations to explore various touch interaction methods: tapping, crossing (drawing a line through targets), exiting (as crossing with targets on the edges of the device), directional gesturing (on blank screen with no targets, participants draw a line in the desired direction). Guerreiro et al. (2010a) used evaluation methods and analysis reminiscent of Y. S. Park et al. (2008) however they reported optimal target sizes of at least 12mm as opposed to the 9.6mm recommended by others such as (S. Lee & Zhai, 2009; Parhi et al., 2006; Y. S. Park et al., 2008).

While Guerreiro et al. (2010b) found that their participants preferred *tapping* interactions, the results showed that crossing, exiting and directional gestures were all suitable for motor-impaired users. They also noted that some directional gestures

produced poorer performance than others. They suggested that interfaces should favour vertical and horizontal over diagonal directions for this particular user group. These findings echo the earlier work by Froehlich et al. (2007) and Wobbrock et al. (2003) who found that the edges provide added stability for target acquisition with stylus touch input by users with motor impairments.

The *Walking User Interfaces* proposed by Kane et al. (2008) adopted a widget scaling technique similar to (Guerreiro, Nicolau, Jorge, & Gonçalves, 2010a) to improve touchscreen device interactions made by able-bodied users while walking (Figure 2.3), highlighting this relationship between user abilities and situational impairments, as subsequently demonstrated by Nicolau (2013).

Henze et al. (2012) investigated both visual and non-visual adaptations to touchscreen interfaces to address target acquisition tasks. Applying touch offset models based on individuals' touch behaviours, the authors evaluated the effects of providing users with the visual feedback of a red dot showing the offset interpretation of their touch, and non-visual methods which simply applied the touch offset model to typing behaviour. The visual feedback interface is shown in Figure 2.5. Hezne et al. (2012) reported that with no scaling of targets, using only the touch offset models and visual feedback of touch location they were able to reduce typing error rates by 18.3%.

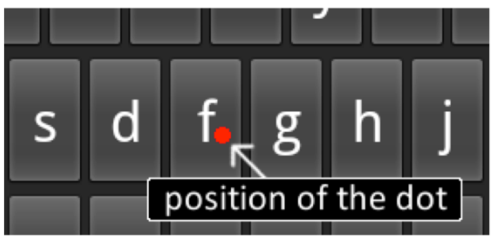

Figure 2.5 Keyboard providing visual feedback of users' touch location when typing the letter "f", extracted from (Henze et al., 2012).

Personalised input (Findlater & Wobbrock, 2012), used similar techniques to Henze et al. (2012), creating touch models of users' interactions with keyboard input on large touchscreen devices. Using visual and non-visual techniques the personalised input tailored keyboard layouts to match the typing behaviours of the individual, shown in Figure 2.6. However, Findlater and Wobbrock (2012) found that while the non-visual adaptations improved typing speeds when compared with the conventional keyboard, the visual adaptations provided no improvement. Furthermore, as shown in the subjective measures, participants preferred the non-visual adaptations. Findlater and Wobbrock (2012) observed the typing speeds decreasing when participants began using the visual adaptive interface condition. They hypothesised this could be the result of an increased cognitive load due to the frequently changing interface.

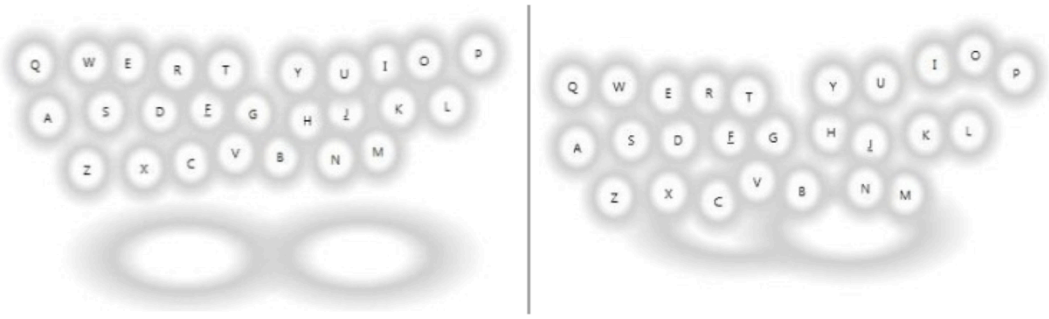

Figure 2.6 Personalized Input keyboard layouts generated for two users, extracted from (Findlater & Wobbrock, 2012).

An important challenge relating to adaptive interfaces is the need for control. While the aforementioned works have applied interface adaptations specific to each application or task, the *Dynamic Keyboard* (Trewin, 2004) and *ACCESS* framework (Heron, Hanson, & Ricketts, 2013) performed adaptations to system wide

configurations. The *Dynamic Keyboard* continuously monitored typing behaviour and adjusted the keyboard filters for all user interactions with the computer.. Similarly, the *ACCESS* framework would continually monitored users' interactions to assess if their needs were not being met with the current configuration settings of the system's input and output, at which point the framework would notify the user and present possible configuration changes. Both *Dynamic Keyboard* and *ACCESS* make adjustments to the configurations within the operating system settings panels and are therefore, present and for all interactions and applications. Furthermore, by altering the OS configuration settings that can be accessed independent of the adaptation systems, they allow users to veto or revert any adaptations and changes made on their behalf and thus, provide the end user with greater control over their interaction experience. Furthermore, because the *Dynamic Keyboard* and *ACCESS* framework examine low-level events beneath the application layer, they are able to continuously monitor and measure user performance without the need for calibration exercises.

2.2 Discussion of Challenges

This section presents a review and discussion of several projects working with the ability-based design area, providing a comparison of these works on the following criteria: *measurement method, scope of method, research environment, adaptability* and *measurement subject.* The section will explain and discuss each comparison criteria in detail. A complete overview of the compared works is presented in Table 2.1 below.

2.2.1 Measurement Method

Measurement method refers to the technique used to capture accurate measurements of the user's interaction abilities. There are a number of methods to acquire user related measurements including: *user preferences* captured through configuration panels or system prompts, a somewhat out-dated approach less applicable to ability-based solutions; *calibration tasks* the most commonly adopted measurement method (Table 2.1), requiring users to undergo a series of tests performing actions such as mouse clicks or onscreen taps to collect clean representations of actions for modelling; and finally *natural interaction,* considerably less common than calibration tasks as it is much more complex to work with and extract clean models of interactions. Natural interaction techniques capture user measurements in unobtrusive background methods allowing users to engage with the technologies in an unaffected way, while the calibration task approach subjects the users to periodic interruptions to acquire updated measurements of their abilities.

The *Dynamic Keyboard* (Trewin, 2004) is an example of the natural interaction measurement method, leveraging the interaction patterns between keyboard and mouse usage to identify periods of intended keyboard typing, then monitoring users' keystrokes to recognise possible typing difficulties and user abilities. The difficulty for such systems is identifying intent: *Did the user really mean to do that?* Gajos et al. (2012) investigated this challenge with mouse pointer interactions, using calibration tasks to capture baseline measurements of user performance and classify interactions that occurred during periods of distraction to distinguish them from intended user interactions.

Natural interaction measurement methods are to be strongly preferred, particularly when considering users with abilities that are prone to large fluctuations and change (Hurst, Mankoff, & Hudson, 2008b).

2.2.2 Scope of Method

Scope of Method relates to the analysis technique applied to the measured data, whether it is specific to a particular task application or whether it is generic enough to apply to measurement data in multiple contexts. In most cases the analysis of data must leverage task knowledge to infer intent of user interactions and improve accuracy of user measurements (Table 2.1). Typically the method will involve the user interacting with either a single or highlighted element, allowing the authors to automatically infer that the user's intention was to perform that action, such as clicking (Gajos et al., 2012; 2007; Hurst, Hudson, Mankoff, & Trewin, 2008a; Hurst, Mankoff, & Hudson, 2008b; Trewin et al., 2006) or tapping onscreen targets (Findlater & Wobbrock, 2012; Guerreiro, Nicolau, Jorge, & Gonçalves, 2010a; Y. S. Park & Han, 2010). As a result of this, such systems require the inclusion of a calibration task to elicit the user performance measurements.

Hurst et al. (2010) presented a method using computer vision techniques to locate and identify targets. The common interaction behaviour across most applications when responding to mouse clicks is to change the state of the button or target with some form of highlight or selection visualisation. Collecting boxed grabs comprising the 300x300 pixel area around the cursor before and after a mouse click event, Hurst et al. (2010) were able to extract features from the screen shots to identify the interface element the user clicked, as well as properties relating to the dimensions of

the target, and the relative distances between the click location and target. This approach provided data that could be analysed using the traditional mouse pointer measurements for offsets, movements, target crossing and unintentional clicks. Leveraging these common interaction behaviours (visual state changes of button clicks) was key to the success of the method applied within (Hurst et al., 2010), and provides the basic principles to build upon in order to strengthen this technique.

2.2.3 Research Environment

Research Environment describes the context for which the user measurements are captured. Typically speaking, there are two categories of research environment: laboratory and real world (Table 2.1). However research may also synthesise real world conditions (Kane et al., 2008) to maintain a level of control that is not possible in the real world. The inclusion of real world evaluation and user measurements are vital for the consideration of situational impairments and variable health conditions (Nicolau, 2013).

Controlled, task-specific laboratory studies are often adopted within the field of HCI due to the high levels of control they offer. However this control comes at the price of limiting the understanding of how users interact with systems in a real world setting, over time, and while being unobserved. In-situ user evaluations can illuminate real world behaviours and expose challenges and barriers that would have never otherwise been identified. Again the force inhibiting the widespread adoption of real world evaluations and measurements is understanding user *intent*. As a result many researchers (Gajos et al., 2012; Goel et al., 2012; Henze, Rukzio, & Boll, 2011; Hurst, Mankoff, & Hudson, 2008b) have opted for the use of the semantically

meaningless calibration tasks to obtain user measurements. However, Trewin (2004) and Hurst et al. (2010) have demonstrated that it is possible to conduct these evaluations outside of the controlled laboratory, capturing natural interactions.

2.2.4 Adaptability

Adaptability refers to the system's ability and approach to adaptation: *none,* in which the system provides users with a static interface and the work concentrates on collecting measurements of performance; *self-adaptive,* in which the adaptation process is entirely based on measurements and does not allow for user control; or *user-adaptive,* in which adaptations are based on the measurements but can be altered by users.

2.2.5 Measurement Subject

Measurement Subject is used to describe the basis for a system's measurements or interface adaptations, split into two categories: *group* and *individual.* Works applying group level measurements or interface adaptations are not fully consistent with the ability-based design ethos. However, their discoveries and methods are fundamental to the development of design solutions that do consider the individual nature of abilities and situations.

Related Work	Measurement method	Scope of Method	Research Environment	Measurement Subject	Adaptability
SUPPLE (Gajos et al., 2007; 2008)	Calibration task	Task	Laboratory	Individual	Self-Adaptive
Steady Clicks (Trewin et al., 2006)	Calibration task	Task	Laboratory	Group	None
ICD-2 (Flatla & Gutwin, 2011)	Calibration task	Task	Laboratory	Individual	Self-Adaptive
Automatic Mouse Performance Detection (Hurst, Hudson, Mankoff, & Trewin, 2008a)	Calibration task	Task	Laboratory	Group	None
ACCESS Framework (Heron et al., 2013)	Calibration task	Task	Laboratory	Individual	User-Adaptive
Touch Input for Tetraplegics (Guerreiro, Nicolau, Jorge, & Gonçalves, 2010a)	Calibration task	Task	Laboratory	Group	None
Design for Touchscreen Target Selection (Y. S. Park & Han, 2010)	Calibration task	Task	Laboratory	Group	None
Understanding Pointing problems (Hurst, Mankoff, & Hudson, 2008b)	Calibration task	Task	Real World	Individual	None
Identifying Target Intent (Hurst et al., 2010)	Natural Interactions	System	Real World	Group	None
In Situ Pointing Performance (Gajos et al., 2012)	Calibration task	System	Real World	Individual	None
Personalized Input (Findlater & Wobbrock, 2012)	Calibration task	Task	Laboratory	Individual	Self-Adaptive
Dynamic Keyboard (Trewin, 2004; Trewin & Pain, 1997)	Natural Interactions	System	Real World	Individual	User-Adaptive

Table 2.1 Overview of reviewed papers compared within the related work discussion section

2.3 Physical and Visual Access Problems of Touchscreens

Based on the related works, the major challenges and barriers to access of touchscreen devices by people with visual and or motor impairments relate to the input methods of the technology. Specifically these access problems are around target acquisition, the proposed solutions include tailored interface layouts (Gajos et al., 2007; 2008); target scaling (Guerreiro, Nicolau, Jorge, & Gonçalves, 2010a; Y. S. Park & Han, 2010) and personalized touch offset models (Henze et al., 2011; Findlater & Wobbrock, 2012) and gesture recognisers (Trewin, S., Swart, C., & Pettick, D., 2013). However, many of the previously proposed solutions target a specific disability or stereotypical user group, failing to address the diverse range of motor and visual abilities within these populations.

2.4 Summary

Table 2.1 provides an overview of papers discussed in this chapter, describing the work in the context of the discussed key parameters for adaptation. This chapter has described the influential related works that have helped to shape the journey and outcomes of this thesis. It began with a discussion of ability-based design, the conceptual method selected by this research for its realistic goals and logical approach to developing accessible technologies. The chapter presented the existing barriers and challenges to achieving ability-based systems, discussing approaches to adaptation that inspired the work carried out within this thesis.

Chapter 3. Preliminary Research

The objective of this thesis is to investigate the barriers to touchscreen technologies experienced by individuals with a diverse range of motor and visual abilities. Primarily, the focus of the research is to improve the accuracy of user models for people with fluctuating abilities.

Chapter Two framed the conceptual underpinnings of this thesis in terms of adaptive technologies, then presented a review of the related work and a discussion of the widely adopted approaches employed to address barriers to access of technologies. This chapter begins with a review of work on touch screen interactions. It then reports on the initial phase of the research, which aims to substantiate an understanding of the range of user interaction behaviours with mobile touchscreen devices through observations. The purpose of this initial study was also to inform the technological approach of the research and to establish methods of capturing user data using both the individual's preferences and interaction characteristics.

Presented is an evaluation conducted with older adult participants using a mobile touchscreen device in a way-finding context. The rationale for this study was to observe the initial impressions and interaction behaviours of the users to identify the challenges and barriers to use.

3.5 Understanding Touchscreen Interactions

Target acquisition with mobile touchscreens is a common obstacle for all users due to finger occlusion or the "fat finger" problem (Vogel & Baudisch, 2007). With most mobile devices, phones and tablets being scaled to fit comfortably into a handbag or

trouser pocket, the current screen sizes rarely exceed 13cm (phones) and 26cm (tablets). The impact of these sizes is that they force constraints on the maximum dimensions an individual target can occupy. To compound the problem, the size of the human index finger (ranging from 15.5mm to 18.2mm) in addition to its rounded shape, makes it less than optimal for selecting small targets. This imposes constraints on the minimum dimensions of interface elements intended for interaction. Parhi et al., (2006) conducted user evaluations investigating the effect of target sizes with respect to *discrete* (menu selections) and *serial* (entering text on a keypad) target acquisitions made by young able-bodied users. Participants, in a standing position, held the device in one hand while hitting targets with the thumb of the same hand. The researchers controlled the size of the targets, and reported optimal target sizes of 9.2mm and 9.6mm for *discrete* and *serial* interactions. (S. Lee & Zhai, 2009) and (Y. S. Park & Han, 2010) later confirmed these minimum size recommendations of (~10mm) when using similar study designs measuring serial touch behaviours. Where (Parhi et al., 2006) used targets with equal proportion, Lee et al. (2009) further explored the effects of the target size in both portrait (4.9x8.3mm, narrow) and landscape (7.5x6.5mm, wide) keyboards of the iPhone, which showed reduced input speed and increased targeting errors for the narrow input condition. However, the nature of the targeting errors is not discussed with regard to the targeting offsets, due to the limitations of the study apparatus. Lee et al. (2009) used the device default "off the shelf" keypad for the user evaluation, and performed keystroke analysis based on the recorded observations. In contrast, the apparatus and study design used by Park and Han (2010) allowed these types of errors to be captured by the device and reported. Through programmatically recording the users' touch input locations

in relation to the 5x5 grid of 4mm, 7mm and 10mm targets, Park and Han used similar methods to (Parhi et al., 2006), but adopted a seated position using one-handed thumb touch input to hit the on-screen targets. (Y. S. Park & Han, 2010) provided a more rigorous analysis of the touch inputs, exploring the distribution of touch errors, success rates and touch convenience using the 5x5 regions, as opposed to the 3x3 adopted in (Parhi et al., 2006). Park and Han's (2010) inspection of the touch behaviours using the 5x5 regions uncovered touch error offsets in both the *x* and *y axes*. Moreover, they proposed corrective offset values of the device-sensed touch locations for both axes and were able to significantly improve the success rates of target acquisitions. Park and Han (2010) proposed touch offset shifts of -2,-3 pixels in the x, y axis (making a shift of 1.4mm in the real world) for their user population.

3.5.1 Motor Impairments and Touch

All of the above user studies were carried out with young able-bodied participants, some of who regularly used mobile touchscreen devices. However, Guerreiro et al., (2010a) conducted a comparable evaluation to Parhi et al. (2006), Lee et al. (2009) and Park and Han (2010), investigating touch inputs by participants with motor-impairments. The evaluation went beyond target acquisition with *tapping* interactions and explored the ability of tetraplegic users to perform tapping, crossing (drawing a line through targets), exiting (as crossing with targets on the edges of the device), and directional gesturing (blank screen with no targets, participants draw a line in the desired direction). Applying an equivalent analysis to that in (Y. S. Park & Han, 2010), the tapping results identify a need for larger targets of at least 12mm for users with motor-impairments. Furthermore, while the success rate distributions

across the 5x5 grids share agreement for more accurate target acquisitions in the centres of the screens, the regions around the edges of the device show extremely conflicting views. It could be argued that these variations in the distributions are the result of the differences within the study design. As opposed to adopting a seating position (Y. S. Park & Han, 2010), participants were encouraged to place the device in a comfortable position (including placing them on tables or armrests) and use any part of their hand to interact with the device. This allowed them to behave more naturally with the device.

3.5.2 Target Perception and Touch

Holz & Baudisch (2010) have carried out extensive investigations to identify the inaccuracies in touch precision that may be attributed to the discrepancies in the perception of human touch locations and the device interpretations from the contact areas. They conducted a series of user studies to explore the rationale behind users' *targeting procedures* through participant interviews and trials using low fidelity paper targets, digital track-pads and cameras to capture participants' targeting efforts across multiple finger orientations and postures. The work evaluated new touch models based on users' perceived input, and six finger feature specific models designed to correct for the targeting offsets, all of which produced lower error rates than traditional contact models. The greatest results were achieved when using the model built from the user's *projected centre* point of the fingernail. In this condition, targeting offsets were reduced to 1.6mm (40% of the magnitude used with traditional contact models). This suggests that this is the *targeting procedure* applied when users interact with touchscreens. However these models rely on knowledge of finger features including the base, tip and the sides of finger nails; information that

cannot currently be acquired through capacitive sensing technologies used within mobile devices.

Collectively all of these studies are limited by the same weakness in that they failed to consider that their constrained one-handed thumb interaction method might not be the most natural or optimal configuration for all of their participants. Furthermore, they all take place within a restrictive laboratory setting, far from the environment usually associated with mobile devices. Through learning from the limitations of these studies proceeding, this current user study aims to contribute to addressing a gap in knowledge as it investigates the natural usage behaviours of touchscreens by users within real-world tasks.

3.6 User Study

This section presents an exploratory study carried out with four older adults. The objective of this evaluation was to identify the characteristics and behaviours of users when interacting with mobile touchscreens, to identify the similarities and differences between users and to better define and scope the challenges with touchscreen interactions. Gregor and Newell (2001) discussed the dynamic nature of the human species, calling attention to the various stages of change in a lifetime. In particular they highlighted the decline in cognitive, physical and sensory abilities over time. Therefore, the older adult population embody a diverse set of characteristics and capabilities, many of which overlap with people with visual and motor impairments. Furthermore, this study aims to inform the research direction and data collection methods of this work.

3.6.1 Participants

Four older adults were approached and recruited through the School of Computing's User Centre, which is a computer drop-in centre for older adults to learn about technologies. The inclusion / exclusion criteria for this study required that the older adults had low visual or, and motor abilities to participate, each participate self-reported as meeting these requirements. The group consisted of two female and two male participants aged between 60 and 86 years (M= 72, SD= 12.11) (Table 3.1). All participants owned and regularly used a mobile phone, none of which were smartphones. None of the participants had used a mobile touchscreen before but each participant reported using the self-serve touchscreens at supermarkets. See appendix 1 for information sheet and consent forms.

Participant	Age	Gender	Impairment	Accommodation
P1	60	Female	None	N/a
P2	86	Female	Macular Degeneration (left eye), Loss of hearing (left ear)	Wears varifocal glasses, hearing aid
P3	64	Male	None	N/a
P4	78	Male	Loss of vision	Wears reading glasses

Table 3.1 Overview of participant information: age, gender, impairment and accommodations.

3.6.2 Apparatus

The mobile device selected for this study was the second generation Apple iPod Touch (Figure 3.1) running iOS 3.0, full technical specifications are detailed in Table 3.2.

	iPod Touch
RAM	128MB
CPU	533MHz
Network	Wifi / Bluetooth
Camera	N/A
Battery	739 mA-h
Weight	115 g
Microphone	Yes
Accelerometer	3-Axis
Vibration Motor	N/A
Screen Resolution	320x480
Pixels per inch	163
Screen Dimensions	74 mm (H) 49 mm (W)
Device Dimensions	110 mm (H) 61.8 mm (W) 8.5 mm (D)
Operating System	iOS 3.0

Table 3.2 Summary of second generation iPod Touch hardware specifications (Wikipedia, n.d.)

A prototype indoor navigation application was produced using JavaServer Pages (JSP) and MySQL database. This database contained a graph representation of the School of Computing building, and supported navigation queries in the form of an origin and destination location and optional route parameters. Each room or passageway was defined and stored as a node within the graph, and vertices were used to represent the connections between rooms and passageways within the physical space. In addition to storing the connected locations, nodes could also have

supplementary media items such as images. The indoor navigation prototype application could request way-finding instructions between two locations and specify additional constraint parameters for the route, for example, to provide a route that avoids stairs. Origin and destination locations could be selected from a list of all available rooms or people within the building, while route constraints needed to be defined within the settings panel of the application. Once the origin and destination locations were selected, the indoor navigation application would traverse the graph to identify an appropriate route for the request, and return the user a series of navigational instructions that included textual directions and media elements where available, as shown in Figure 3.1. All the computation was performed server-side and an HTML page was returned to the user's mobile device. Whilst the HTML prototype was highly portable, the loading times of the pages were very inconsistent. To resolve this issue, web service access was added and the prototype was embedded with a Simple Object Access Protocol (SOAP) client to retrieve the navigation results. Code was added to the indoor navigation application to log user interactions, tracking button selections and page loads. The interaction logs were collected via the JSP web services, requiring the application to be in constant connection with an active WiFi network.

The indoor navigation application was designed in accordance with the iOS design guidelines (Apple, 2009), ensuring the correct interface elements and layouts were used and minimum target sizes all conformed with the iOS guidelines. Each button was built using the standard UIButton interface element (with additional styles applied) and therefore, responded as any other application available on iOS would be expected to respond.

The navigation interface presented to the user is detailed in Figure 3.1. The participant could navigate between the way-finding instructions using the previous and next buttons (F and H in Figure 3.1). When the previous or next buttons were selected the application would send a SOAP request for the corresponding content, loading the related image and text into the interface. Due to the device's lack of vibration motor, no tactile feedback was provided to participants. However, the application did emit a beep when a tap gesture was recognised to inform the participant of a successful action. The second generation iPod was never embedded with the Apple VoiceOver screen reader software. Thus, in order to provide participants with text-to-speech transcriptions the application was embedded with code to communicate with the Google Translate[1] service. When participants pressed the audio button (G) the application would beep then make a request to the Google Translate service, which would then return an .mp3 file of the spoken text. Finally, participants could hide and show the text instruction overlay by tapping the show/hide button (C). This allowed participants to view the full image, particularly useful when the instructions spanned multiple lines and occluded much of the image. The interface components would only respond to a *single finger tap* gesture.

1 http://translate.google.com

Figure 3.1 The 2nd generation iPod Touch running the Indoor Navigation app.

3.6.3 Procedure

Participants were informed that the rationale for the study was to investigate the indoor navigation tool, and the personalised navigation interfaces. The study also aimed to investigate the broader interaction behaviours of the users when using a touchscreen mobile device in situational context. The study consisted of a single session lasting 30-45 minutes, composed of three elements: an initial interview with the researcher, two way-finding tasks using the mobile app, and a final discussion for debriefing and feedback.

3.6.3.1 Initial interview

The initial interview collected data about each participant's mobile phone usage and touchscreen experience. The researcher provided the participants with a short tutorial demonstration of the iPod touch, explaining that to interact with the onscreen targets

the user must *touch the item to select it*. A conscious decision was made not to explain the touch sensing mechanisms thus allowing participants to explore the interactions naturally: *What makes a touch? Is it timing, pressure? Is my finger too big, too small?*

Within this initial interview participants also provided preferences (Figure 3.2) for how they received the navigation information. They selected one or a combination of interface modalities: *text, images, audio transcriptions*, and their preferences for the navigation route itself. Their preference for navigation included, for example, whether they needed to avoid stairs when moving between floors.

Figure 3.2 iPod Touch showing the Indoor Navigation preference settings screen.

3.6.3.2 Way-finding TasksUsing the Mobile App.

All of the participants regularly visited the User Centre within the department. However, none of them had ever explored the building beyond the ground floor. For

this reason each of the way-finding tasks required participants to navigate from the entrance of the building and locate offices in the upper levels of the building. Each participant was asked to complete two way-finding tasks using only the navigation instructions provided by the indoor navigation app. The application presented the user with a single instruction using *text, image* and *audio* modalities depending on individual preferences set previously, shown in Figure 3.1. No indoor localisation was provided by the application, users were required to match the descriptions or images to their current location and respond to the instructions accordingly. The researcher walked with the participants during the way-finding tasks to make observations, but provided no assistance with regards to touchscreen usage or the way-finding tasks.

3.6.3.3 Final Discussion

Once both indoor way-finding tasks were completed, each participant returned to the laboratory with the researcher for a study debrief and informal discussion about their experience. During this discussion participants were free to comment on the study, application or touchscreen technology, while also allowing the researcher to ask questions relating to specific behaviours or instances within the tasks to help understand the participant's intentions.

3.7 Results

The primary goal of this user evaluation was to observe the touchscreen interactions, to identify the behaviours that are common and unique across users. Secondly, this study aimed to determine the barriers and challenges of touchscreen interactions by users with lower levels of motor and visual ability. Finally, the objective was to

inform the direction and technological approach of this dissertation to define methods of capturing performance measurements. The results are discussed in relation to the user implications and technical requirements of the approach.

3.7.1 User Implications

First, the study has implications for users' ability to specify initial configurations for software. Configuration screens are commonly presented to users upon the initial launch of a device or an application as a quick way to establish some baseline user preferences. As noted by Trewin (2000), in order to use these screens accurately, users must understand the best settings for use. The current evaluation suggested that the user may not always be able to provide this information due to his/her lack of knowledge or experience with the very application he/she is attempting to configure. For example, while users may be aware of their own limitations, they may be unable to translate this knowledge to appropriately configure software. As a case in point, one of the participants in this study understood a vision limitation, *"I need my glasses for reading only" (P4).* However, when asked *"Would you like audio transcriptions of the navigation instructions?"* he replied *"I'm not sure; how big is the text going to be?"*

Similarly, *P2* commented on the screen contrast as being *"Ok sitting here, but I couldn't always make out the instructions on the screen".* This statement highlights the differences between the environment within the laboratory where the preferences were set, and the conditions the participants were exposed to during the way-finding task. It is important to note that these were indoor way-finding tasks only, greater

extremes might have been identified had participants also been asked to explore outdoor environments with the device.

A second implication of this study was in terms of defining touch. Participants were intentionally not informed of the parameters that defined a successful touch in order for the researcher to better understand their perceived requirements and individual touch behaviours. While none of the participants had ever used mobile touchscreen devices before, they all had prior experience with touchscreen self-service kiosks, which used resistive touch sensing. Resistive touchscreens require a reasonable amount of pressure to allow the two layers within the monitor to contact and initiate the touch. Since the participants were familiar with these kiosks, they made the assumption that the mobile touchscreen behaved in the same way, requiring the appropriate amount of pressure to action a touch. When they were asked to discuss their experience with the mobile application, two of them commented on the device not responding *"so I pushed it again harder for longer and it seemed to work fine" (P2)*. *P4* used a similar strategy, *"You have to press it really hard to hit the buttons"*. While participants *P2* and *P4* were applying increase amounts of pressure, *P1* recognised that the device did not require additional force and that a light touch was sufficient to make selections *"It is much quicker* (more responsive) *than the ones* (touchscreens) *at the shops"*. The experimental apparatus did not capture unsuccessful touches as they did not generate application actions such as new page views. However, the participants all applied a similar strategy in the event of an unrecognised tap of pushing the same target again. While *P2* and *P4* also talked about applying more pressure, all of the participants suggested trying again and holding the button for longer. Potentially this strategy could be applied to identify

instances were the device fails to recognise an intentional input, helping to refine the parameters of the recognisers.

Thirdly, this initial work can help define touches in terms of duration. Previous studies investigated users' abilities to successfully select touchscreen targets of various sizes (S. Lee & Zhai, 2009; Parhi et al., 2006; Y. S. Park & Han, 2010) and using different interaction methods (Guerreiro, Nicolau, Jorge, & Gonçalves, 2010a). However, they did not comment on the timings and durations of these interactions. While the indoor navigation application was not logging the durations of the users' touches, it was both observed during the study and reported by the participants that timings were a factor of touches. *P2* and *P4* opted for the longer firm touch method, *P1* and *P3* used touches that appeared to only just make contact with the screen for an instant.

3.7.2 Technical Requirements

The indoor navigation application used within this study requires connectivity to an active WiFi network at all times in order to receive navigation instructions and store user interactions back to the web services. The WiFi network canvased the entire building, although signal strengths varied throughout and in some cases the connection was lost completely. Because the application relied on an active network, connection to retrieve instructions or request audio transcriptions, the participants occasionally experienced delays during the evaluation causing them to misinterpret the instructions. For example, *P3* was walking along the corridor while waiting on the instructions loading, and received the message *"walk down the corridor; take the first door on the right"*. However, by the time *P3* received this instruction he had

already walked beyond the first door on the right and therefore took the second door and deviated from the route. At that point he was unable to re-orientate himself, and did not complete the task. Similar disruptions were caused by the data logging mechanisms. Since all data was logged directly to the remote server via web services, any delays within these calls or responses affected the user interface and their experience.

Also in relation to the connectivity issues, interface feedback was affected in other ways. The application suffered from variable loading times where some instructions would load instantly, while others would take several seconds. The participants perceived these issues not as a problem of the device, but rather an issue with their own level of touch, and as a result would alter their technique and select the button again. When the participants touched *down* on a button, a highlighted state was activated and this was deactivated on touch *up*. Upon a successful tap the device played a beep sound to confirm this interaction. However, the device volume was set again during the initial preference setting interview in the quiet laboratory. Once the participants began the way-finding task they were exposed to real world ambient noises and fluctuations, with the result that *P2* and *P3* reported not being able to clearly hear any beeps or the audio transcriptions from the device during the way-finding tasks.

Finally, the experimental application recorded interactions at button level, meaning that the log files detailed the buttons that were selected and the corresponding actions such as page loads, or playing audio transcriptions. These log files allowed the researcher to understand how the device responded to the behaviours observed

during the study, but they did not provide enough detail to accurately describe the individual differences within users' touch interactions. All data were captured through the overwritten methods within the UIButton controls. These components have access to finer grain details than recorded within this evaluation. However, there is a need for local storage and periodical synchronisation of user data to reduce the impact on network demands and device response behaviours.

3.8 Summary

Previous investigations of touchscreen behaviours enforced restrictive environmental and interaction constraints, presenting participants with abstract calibration tasks far from the real world applications and usage situations associated with these technologies. This preliminary evaluation explored the individual characteristics of touch interactions set within the context of a real world application. Using a stimulus application developed in alignment with the current design guidelines and best practices, the study allowed the researcher to identify the differences and similarities of an individual's interaction perceptions, strategies and behaviours.

As a result of the relaxed interaction constraints, this evaluation uncovered new touchscreen interaction characteristics beyond the target acquisition challenges, demonstrating the various differences of touchscreen interactions. While the limitations of the experimental apparatus did not allow quantitative exploration of these characteristics, the research observations were supported with participant feedback suggesting the need for touchscreens capable of responding to differences in gesture timings. Furthermore, this study identified the strategies applied by participants to resolve unsuccessful tap gestures. Leveraging these behaviours

would allow the recognition of failed intentional gestures to refine and personalise the parameters that define the gesture's success criteria.

User preferences were collected during an initial interview and applied to the interface prior to the participants beginning the way-finding tasks. All participants found the process of configuring settings difficult given that they had not yet used the application. Participants were also making configuration choices within the controlled laboratory space, where the environmental conditions did not always match those of the building in which they carried out their way-finding tasks. Interactions were affected by lighting conditions and ambient noise that were not present during the initial configuration stage. Therefore, this study has highlighted the need for greater consideration of contextual factors to support the configuration of user interfaces.

Finally, the data logging techniques applied within this study did not provide enough detail for an accurate analysis of the touch characteristics. There is a need for the user input to be collected at a much higher degree of granularity to measure the timing and duration of touch interactions.

3.9 Conclusion

This chapter introduced the exploratory investigation of touchscreen interactions within the context of a real-world application, and was designed to serve as an introductory description of the research conducted within this dissertation. The chapter presented a preliminary user study that aimed to understand the variances in abilities and behaviours of users when interacting with touchscreen devices. User needs and requirements as well as technical implications were identified as a result

of this evaluation, and influenced the focus and approach adopted throughout the remainder of the research work.

Chapter 4. Development of the SUM Framework

The previous chapter described the preliminary study involving four older adults using the indoor navigation touchscreen application. The application was embedded with an interaction data collection layer to log the application interactions during the study session and these were also combined with the researcher's observations. The resulting analysis of these logs and observations helped to identify and define a set of touch interaction characteristics and behaviours to be further investigated.

This chapter describes the design of the first iteration of the Shared User Modelling (SUM) framework, which will be referred to as *version one* throughout this thesis. The chapter begins with a discussion of the challenges of and motivation for using SUM. Following this is an overview of the software architecture for SUM version one. This overview includes details of and the rationale for the data types recorded and the storage structures used. The chapter concludes by considering the intended method for embedding SUM framework into third party applications.

4.1 The Scope of SUM

Shared User Modelling (SUM) framework has been designed to support the user modelling approach of this dissertation. The primary goal of the framework was to provide the necessary methods to capture accurate measurements of user abilities and performance from real-world interactions. Secondly, the framework needed to enable user modelling of the recorded interactions. Finally, SUM had to provide mechanisms to apply user models back into the applications. It was crucial to define

the user modelling structure that would ensure the framework could support the overall approach of this dissertation.

4.1.1 User Models

User-adaptive systems, more recently known as *adaptive systems*, have been a popular area of research for decades, making use of the extensive user modelling work pioneered by researchers such as Allen, Finin and Rich (Allen, 1990; Finin & Drager, 1986; Rich, 1979). Rich (1979) proposed three fundamental considerations for classification of user models: *Are they models of a canonical (typical) user or are they models of individual users? Are they constructed explicitly by the user himself or are they abstracted by the system on the basis of the user's behaviour? Do they contain short-term, highly specific information or longer-term, more general information?* Rich believed that the major differences along these dimensions corresponded to the resulting forms of adaptive systems. Canonical models of users assume a static state, therefore can be defined and embedded directly into the application. Alternatively, modelling individual users requires data gathering mechanisms to support the creation of the user models at the application usage time. Similarly, when considering the selected methods of capturing the user data, the slow to change or static aspects of interaction can be acquired through explicit definitions whereas the quick to change dynamic aspects are better suited to implicit collection methods by the system.

4.1.1.1 Stereotypes and Individual Representation

Models built from stereotypes of users can be agreed and defined during the design stages of an application. By clustering users into groups based on their similarities,

stereotypical profiles can then be used to tailor a system's behaviour in alignment with the profile characteristics. At use time, the actions and behaviours of users are captured and classified into one or more of the stereotypical groups. The conceptual approach of stereotypical user modelling supports the objectives and processes of the universal design movement (Mace, Hardie, & Place, 1991.) encouraging designers to think of specific user abilities and needs ahead of time and defining rules or behaviours to match these user profiles at use time.

In contrast, user models that represent an individual must be constructed at use time, collecting explicit preferences and user information or implicitly inferring attributes and properties regarding the individual from his/her system behaviours. The evidence collected at use time is associated with particular attributes stored within the user model, allowing the adaptive systems to respond to properties and characteristics on an individual basis. These types of user models are supportive of the ability-based design (Wobbrock et al., 2011) approach, and the direction of user models that will be constructed within this body of research.

4.1.1.2 Explicit and Implicit Modelling

Having users explicitly defining their own models provides clean data with a high degree of certainty for the user intent. However, issues can arise with this method as the user can misunderstand configuration options and may not provide the correct information due to a lack of understanding of the resulting configurations, as was observed and reported within the preliminary user studies (Chapter 3). Richards and Hanson (2004) proposed an adaptation solution for web browsing, which provided users with a simple preference dialog to tailor the presentation of web content

complete with a live preview of the modifications. This solution was able to address the challenges surrounding the users' lack of knowledge over defining their interface preferences. Nevertheless, having users define and configure their own preferences is not suited to short-term models, "*if users specified short-term models, they would have time for little else"* (Rich, 1983), and is therefore ill suited to users with variable needs and abilities. Explicit definitions would require exhaustive input from the user in order to continually accommodate changing needs and abilities.

User models constructed from implicit data collection rely on the system's ability to infer new knowledge with a reasonable degree of certainty in the user's actions or intentions. More often than not, the implicit information is captured using methods that are tightly coupled with specific tasks. For example, a workflow within a word processing application might be to format the document. From this task the system might implicitly infer a user's preferences for font typefaces, sizes and contrasts. Similar to the issues of users misinterpreting the system when explicitly providing information, the system can misinterpret user actions and infer false truths. For example, the user might have been formatting the text document for someone else, in which case those properties inferred are not associated with the user.

There is scope for hybrid systems that are capable of leveraging both explicit and implicit data collection methods, implicitly measuring attributes and properties specific to the user then having the user explicitly confirm or deny the resulting inferences.

4.1.1.3 Short-term and Long-term Information

Rich (1979) distinguishes between short-term and long-term models based on the application of the model information: is it specific to the current task or can the information be applied in a more general form, such as defining an individual's domain knowledge? (Rich, 1983). The approach of SUM is to model the current needs and abilities of the user, then provide interface adaptations to better support the interactions at that point. Therefore, SUM uses short-term models to respond quickly to an individual's needs rather than modelling his/her overall understanding or domain knowledge of a particular application.

4.2 Design of SUM

The SUM framework is a software architecture designed to address the fundamental challenges of designing accessible touchscreen interfaces. SUM is intended to be embedded into applications to capture accurate measurements of a user's current abilities and needs, then adapt application interfaces to meet those needs and abilities. The framework is domain and platform independent, and is designed to support custom accessibility/usability adaptations to provide the most suitable interaction experience for a person based on his/her specific needs at that point in time.

In order to reduce the complexities of ability-based interface design, the SUM framework extends and complements the existing application programming interface (API) protocols supplied to developers by the mobile device operating systems (OS). This architecture allows developers to embed the framework with little programming overhead, disruption to design patterns or impact on application performance rates.

SUM provides developers with a lightweight client framework, responsible for capturing the user's interactions via the various device sensors, then storing this data within a local database. The framework is responsible for synchronisation of user data with the remote SUMServer, and contains the necessary methods for handling the resulting user models. This makes the process of creating personalised interface adaptations automatic from the perspective of the developer, and invisible to the users.

4.2.1 Features of SUM

The preliminary user study detailed in Chapter 3 helped to identify the interaction preferences and challenges faced by users with low levels of visual and motor abilities, when using mobile touchscreen devices. To address these challenges the following features were defined for SUM user models and interface adaptations.

Duration, represents the time interval from the instance the user's finger touches the surface (*Touch Begin*) until it is then removed (*Touch Ended*). The user models will allow the minimum and maximum duration parameters of the tap gesture recognisers to be adapted.

Target bounds, contains both the width and height parameters of the interface elements, and is expressed in pixel units as used by the devices. The user models will contain a minimum target bounds for optimal visual representation.

Touch Offsets, capture the interaction offset behaviours of the user when touching the screen. The user models will capture both the horizontal (*X)* and vertical (*Y)*

touch offsets in pixel units, allowing the user's touch inputs to be shifted and corrected for.

Font Size, contains the minimum font size that the user can confidently read from the screen. This property can be used to set the text sizes of interface elements. If the font size results in a larger bounds than defined by the minimum target bounds, then the interface element will use the bounding size required by the font size and text label.

Modality Preferences, represents the user's preference for textual, audio, and visual modalities of interface components. The user models contain scalar values 0.0-1.0, where 0 is a low preference and 1 is a high preference for the particular modality.

4.2.2 Structure

Figure 4.1 shows the high level architecture of SUM, demonstrating the way that the SUM framework supports the reuse of user models between applications using a centralised user model structure. Once the SUMClient (Section 4.2.3) libraries have been embedded into the application, user interactions can be captured and stored locally within the client database (Figure 4.7). Using HTTP requests with the SUMServer (Section 4.2.4), the SUMClient can transfer new user data to be synchronised with the user's SUM model. Similarly, the SUMClient can request specific model attributes to support the interface adaptations.

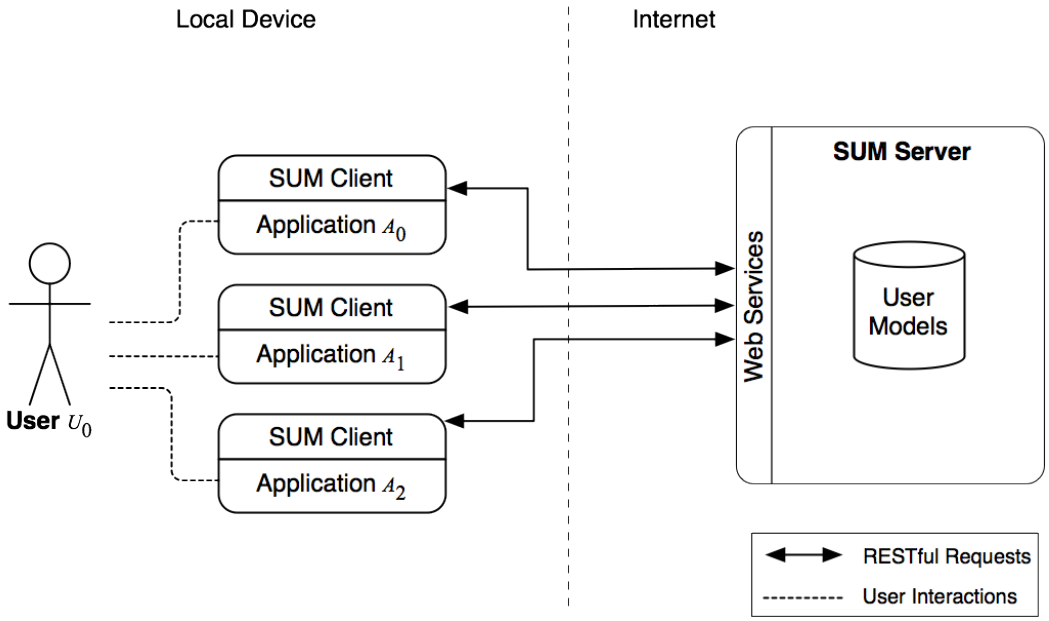

Figure 4.1 UML diagram of the SUM Framework (all versions), illustrating applications accessing the shared user model via RESTful requests with the SUM Web Services

4.2.3 Domain and Platform Independence

The SUM framework has been developed from the ground up, to fully support the exchange of user information between applications and devices. By maintaining a consistent capturing of data, and structure of data storage, between the local application models and the remote server aggregated models, SUM ensures that any and all user data can be mapped from one domain to another. Since the SUMClient framework handles all of the sensor monitoring and storage, user interactions are captured in an identical manner for each application using the framework. As a result, unlike other conventional domain-independent modelling techniques, no model mappings need to be generated for SUM application mappings within the same device platform.

However, due to the many varieties of mobile touchscreen devices, and the various software OS available on the devices, there can be ambiguities in the sensor recordings between the device platforms. These inconsistencies can be the direct result of different hardware properties or sensors, or indirectly due to the individual design of OS software. The result of the variances between device platforms means that SUM is required to use model mapping techniques to ensure that user information is interchangeable across device platforms. However, SUM maintains a hardware profile for each device and OS pairing, providing the required information to map sensor information between platforms. For example, the device *D1* has the following attributes *(pixelWidth:320; pixelHeight:480; mmWidth:76.2; mmHeight:81.28),* device model *D2* similarly had the attributes *(pixelWidth: 1024; pixelHeight: 768; mmWidth: 177.8; mmHeight: 101.6).* These attributes allow the touch data and target information to be normalised and expressed using relative positioning and sizing.

The user models are normalised using the device properties when the data is aggregated in the SUMServer, with the server model representing a canonical model as shown in Figure 4.2 below.

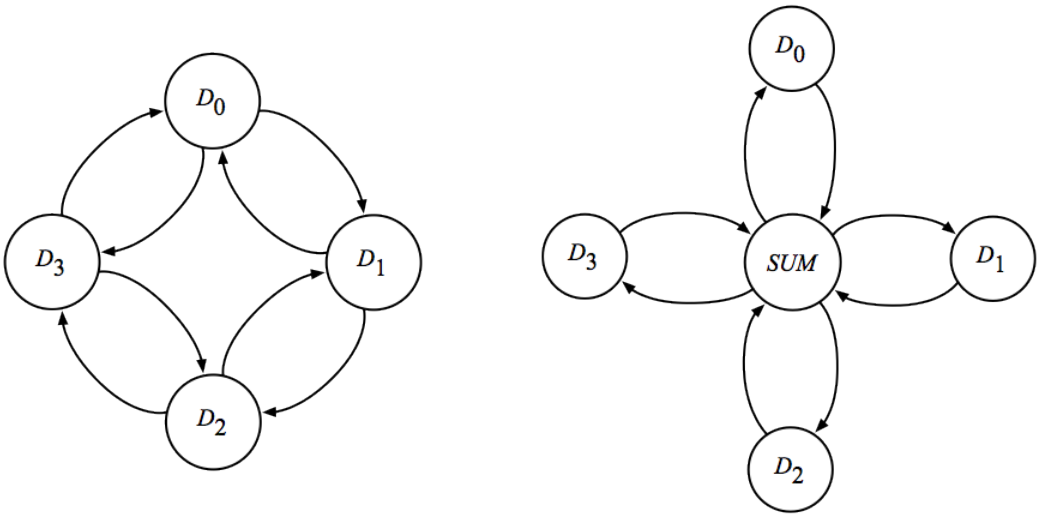

Figure 4.2 Direct model mapping (left) and SUM canonical model mapping (right).

This approach was selected as opposed to the alternative of direct mapping between devices for two reasons. Firstly, the canonical model simplifies the processing structure of the SUM framework – the methods need only be coded to deal with this one type of model. Second, the SUM framework scales for future devices. Using the canonical model structure, adding a new device results in a single bidirectional mapping from device *Dx* to SUM. In contrast, using a direct mapping approach adding a single new device would require bidirectional mapping from *Dx* to all other previously mapped devices. Figure 4.3 illustrates the effects of adding new devices within both mapping approaches.

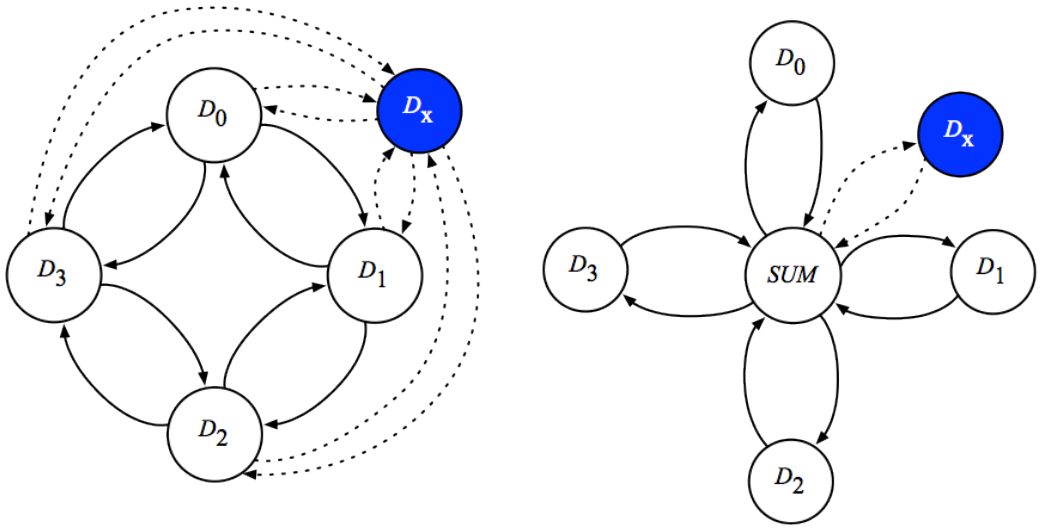

Figure 4.3 Adding a new device, direct mapping (left) and SUM canonical model mapping (right).

The adopted canonical mapping technique also serves as a safeguard to the development of a standard user model structure in the future. Using the model mapping technique previously discussed, the SUM canonical model could be mapped to other user model structures. The process would require an expert to manually define the attribute mappings between SUM and the new model structure just once, which would then allow the interoperability for any previously connected platforms through SUM to the new model structure.

4.2.4 SUMClient

The SUMClient manages interaction data capturing and synchronisation, meaning that developers need only exchange their existing user interface controls with the ones provided through the SUMClient application programming interface (API) to pass interaction data to the framework. This is a simple substitution of the iOS UIControl class with the SUMControl overwritten version, which is coupled with the user model data. These new interface controls will log interaction data and refine the

associated gesture characteristics such as touch duration, target offsets match the abilities of the user.

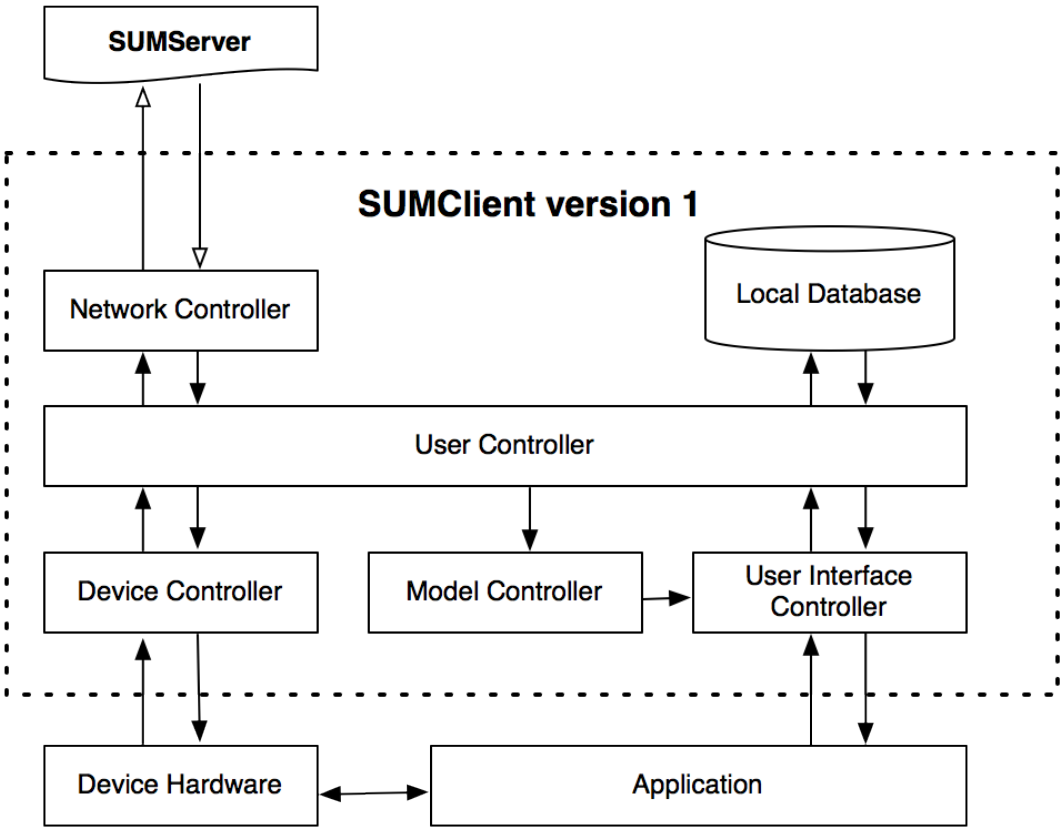

Figure 4.4 SUMClient software architecture showing the communication between internal components

The overwritten UIControls provided through the SUMClient framework automatically add the additional code required to capture and store any touch interactions made with that UIControl. These interface controls respond to the three touch states: *touch begin, touch move* and *touch end.* Since each control is embedded with the code, the framework is able to directly map the user's touch interactions with the specific interface control. Figure 4.5 demonstrates this association, where touch *T1* occurs within interface control *UI4.* The resulting interaction log would be:

```
Touch:{id:1, x:178, y:245, timestamp:1370414769, duration:0.921,
target:{id:4, x:165, y:210, width:140, height:60} }
```

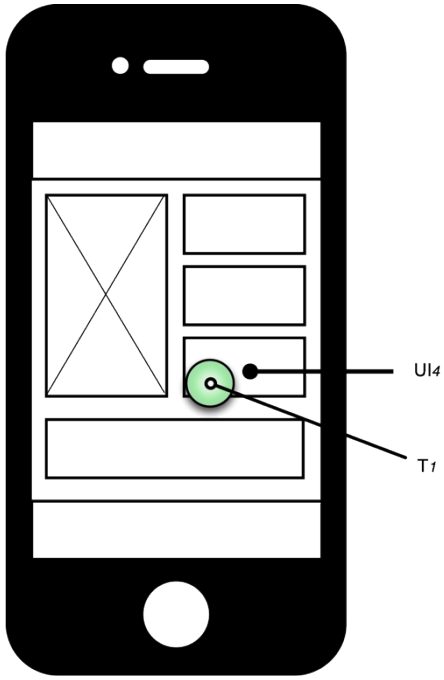

Figure 4.5 Diagram showing the touch recording concept applied by the SUM Framework to an interface.

Measuring the touch interactions at this level of detail allows measurements of *touch offsets, durations* and *time between touches* to be made in relation to target *locations* and *sizes*. Touch offsets are calculated as the distance of the user's touch to the centroid of the target. Duration represents the length of time from the instant the touch-begin event is received, to the time of the touch-end event. SUMClient uses millisecond accuracy of touch durations to identify small variations in timings.

SUMClient uses a mobile device's built-in motion sensors to capture the raw device motion values. To ensure high definition of motion events, data is logged at 100Hz. No processing is performed within the SUMClient on the motion data; it simply measures and stores the magnitude values from the sensor. At the time of development, the sensors output the combined user motion and device orientation within the same value. The iPod touch used within the preliminary research (Chapter

3) contained a tri-axis accelerometer capable of providing measurements of the device movements as shown in Figure 4.6. The raw acceleration values were processed within the SUMServer, applying filters to isolate motion data specific to user movements, and device orientations.

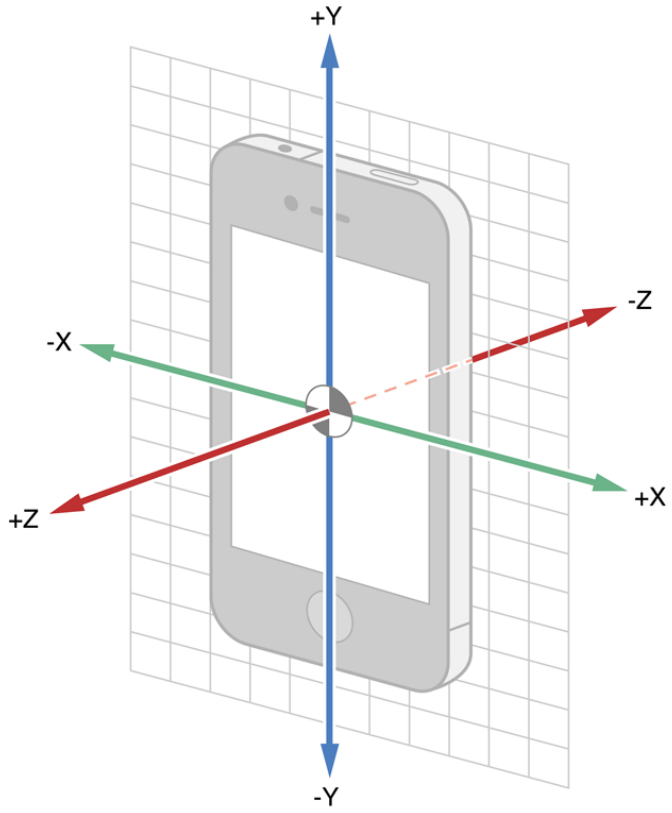

Figure 4.6 Mobile device accelerometer axis relative to the device orientation.

Figure 4.7 shows the high-level table structure of the local database within the SUMClient framework. SUM framework focuses on the accurate measurement and modelling of user interaction abilities and the local database has been designed to support these objectives. The database was developed using SQLite[2], a lightweight SQL database engine that is compatible across the OS of most mobile devices.

[2] http://www.sqlite.org/

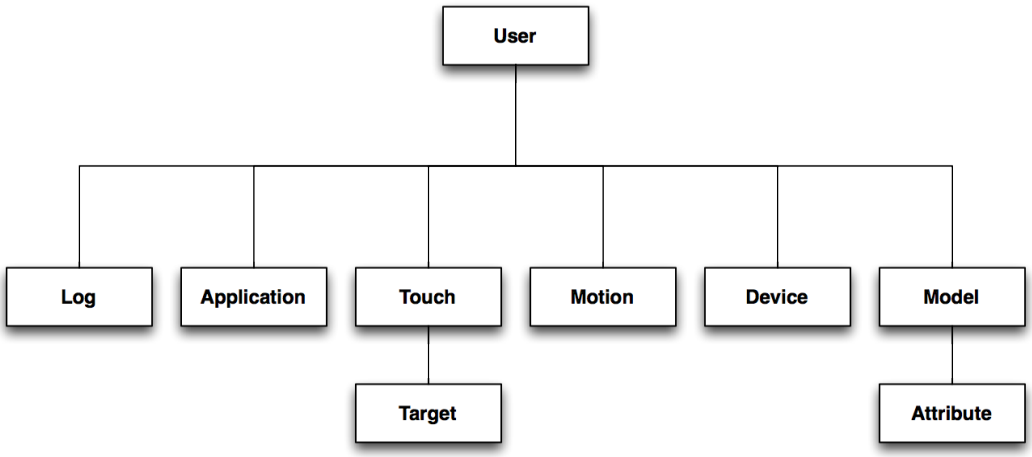

Figure 4.7 UML diagram of the SUM database structure (Version 1) used for capturing user interaction.

User: For authentication purposes users must set up a username and password, at which point they are allocated a unique user identification number. All user measurements and constructed models can then be associated to an individual user and not the device, thus allowing multiple users to share a single device and its applications.

Application: Before an application can access the SUM framework to collect user data and retrieve user models it must first be registered with the SUMServer. Once registered, applications receive a unique application key (*appKey)* which is associated with any data collected through that application. Creating this association allows for closer inspection of user data in relation to the origins of its collection. Furthermore, this is a fundamental component for the implementation of security protocols to allow end users control over the data that can be captured or accessed by a particular application. While these issues are outside the scope of this thesis, it is recognised as an important component to the success of such user modelling systems.

Device: The SUM Framework has been designed as both domain and platform independent. To support this functionality the system needs specific knowledge of the device properties and sensors in order to create the required data mappings from the device values to the normalised SUM model. The device profiles included the following: maximum touches; screen width; screen height; wifi enabled; internet enabled; has accelerometer; has gyroscope; has microphone; has camera and has audio output.

Touch and Target: SUMClient captures touch measurements in relation to the user interface elements that were being interacted with (Figure 4.5). A single tap gesture recorded through SUM would contain the following: *unique id; timestamp; x and y screen locations; duration of gesture; target x and y locations* and *target width and height values.* Capturing the interactions at this level of detail allows SUM to perform similar measurements and analysis techniques as the contemporary projects previously discussed within the scope of SUM (Section 4.1.1).

Motion: Device motion is captured through the built-in accelerometer or gyroscope sensors; SUMClient captures the raw sensor output from each sensor update storing the *x, y, z magnitudes,* and the *timestamp.* The motion magnitudes contain both the device orientation and the user movement data, therefore filters would need to be applied to isolate the specific information of interest. For example, applying a low pass filter can reveal the effects of gravity on the devices accelerometer sensor and applying the accelerometer axis overlays (Figure 4.6) to the motion data provides the device's orientation.

Model and Attribute: SUM has been designed to respond to short-term changes in user behaviour. Consequently, the user does not have a single user model; instead new user models can be generated upon request, pulling newly available interaction evidence to produce a user model that is fitting for the user's current needs and abilities. Each model has a unique identification number linked with the model instance stored within the SUMServer and a timestamp for when the model was created. Then, within the *Attribute* table the model data is stored using the highly flexible *key-value pair* structure, to allow SUM models to handle future attributes beyond the current defined set of properties of interest.

Log: This attribute does not have a direct relationship to user measurements or models. However, the log allows for checkpoints to be placed within the application for debugging. Each log contains a text description and a timestamp. Within the context of this research the log functions were used to gain a finer grain understanding of the specific pages and actions associated with a particular touch. For example, *Page loaded: Contact details - John Smith* therefore any touch interactions after this timestamp took place within the *Contact details* page for *John Smith*.

SUMClient does not perform any data modelling; it is responsible for collecting measurements of touchscreen interactions and relaying this data to the SUMServer. To retrieve a new user model the SUMClient sends an HTTP request to the SUMServer to generate a new user model. SUMServer will then return a JSON string containing the new model for the current user. For example,

```
model:{id:123, timestamp:1370414769, attributes:{
      attribute:{key:Tap_MinDuration,value:.021},
      attribute:{key:Tap_MaxDuration,value:.928},
      attribute:{key:Tap_MinTargetWidth,value:40},
      attribute:{key:Tap_MinTargetHeight,value:32},
      …}}
```

The model is then stored locally and can be accessed via SUMClient API calls to retrieve specific model variables. While the SUM models would allow for any key-value pair to exist, the SUMClient provides a standard set of functions to support interface adaptations and queries consistently across applications avoiding domain specific terms. The initial set of model attributes was selected based on the characteristics identified within the preliminary study (Chapter 3) and related work (Chapter 2). The SUMClient provides accessor methods for the features of SUM (section 4.2.1):

- Minimum and Maximum Duration
- Minimum Target Width and Height
- Target Offsets
- Minimum Font Size
- Preferences for Text, Audio and Images

4.2.5 SUMServer

The SUMServer consists of the SUM modeller, user model database and front facing web services that manage application access, data synchronisation and user model requests (Figure 4.9). To ensure data consistency, only applications embedded with the SUMClient may access the web services. This restriction guarantees that any data provided to the SUM Framework has been collected and maintained in a consistent manner. The design is also supportive of the security and authentication methods (Section 4.3.2). SUMServer has been developed using JSP and MySQL, both of which are supported on most Apache server installations.

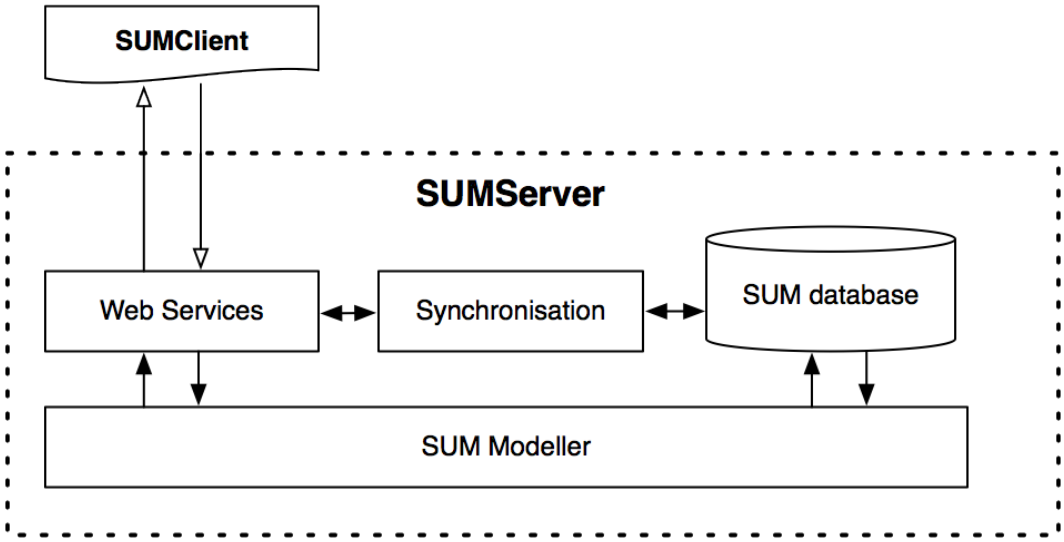

Figure 4.8 SUMServer software architecture showing the communication between internal components.

SUMServer uses a REST architecture, enabling lightweight mechanisms for synchronising user data and requesting user models. The web services conform to the best practices of REST allowing the SUMClient to take advantage of the device's

native REST API if available thereby helping to minimise code duplication and reduce performance impact of the client functions.

Date synchronisation is manually performed through the settings panel embedded within the application using the SUMClient. The researcher can request that the application transfer all local data or make a request for an up to date model for the current user.

A new synchronisation request will create a new session entry associated with the user and return the session's unique identification number. Once the SUMClient receives the session id, it can then parse all of the new user data making the necessary POST requests to add the data with the session id. For example, adding the touch event illustrated in Figure 4.5.

```
URI: jsp.computing.dundee.ac.uk/SUM/touch

Method: POST

Parameters:{session_id:(ID from SUMServer), x:178, y:245,
timestamp:1370414769, duration:0.921, target:{id:4, x:165, y:210,
width:140, height:60} }
```

The SUM modeller does not use any machine learning techniques at present. Due to the lack of detailed interaction measurements from the preliminary study (Chapter 3), the SUM framework uses simple statistical methods to define optimal parameters from the user's previous interactions. SUM modeller creates a Gaussian distribution of the attribute's values then defines bounding parameters using the lower and upper 5th percentiles to identify the range of possible values within the given attribute. Figure 4.9 illustrates the Gaussian distribution for the duration of a tap gesture, successful timings occur within the inner bounds of the distribution.

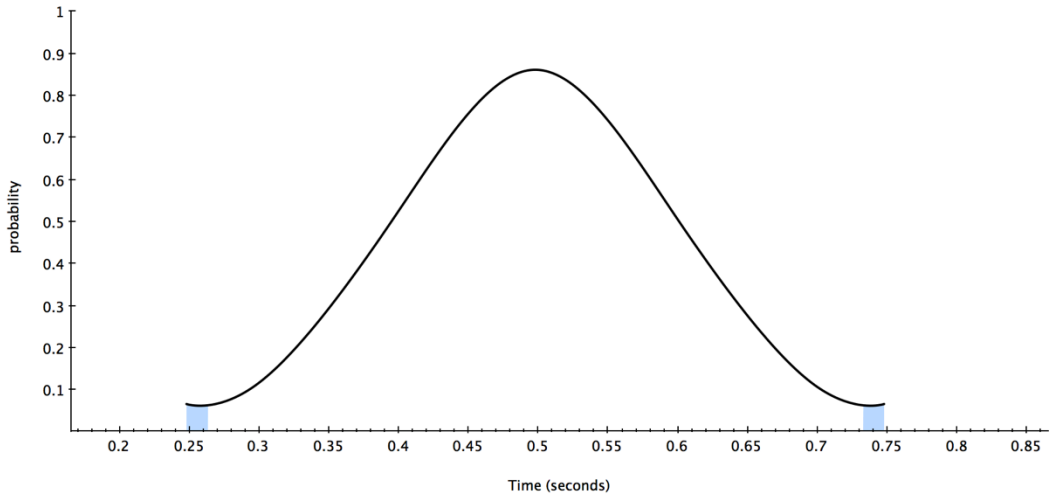

Figure 4.9 Gaussian distribution of touch durations: the highlighted areas represent the 5th and 95th percentile cut off points for the model.

4.3 Building with the SUM Framework

Although the primary goal of this work is to investigate the potential of the SUM approach to user modelling, it was also important to define realistic methods to achieve such models. Therefore, the SUM framework has been designed to match the current mobile development workflows and present a minimal impact on device performance and network activity. Similarly, much of the functionality such as authentication and interface adaptation has been automated to remove the burden for the developers.

4.3.1 Library and API

Designed to minimise disruption of mobile application development workflow, the SUMClient is packaged into a single self-contained static library file for iOS development. Developers can import the static library through the standard development process, and then include the framework within the project by adding `#import SUM/SUMCore.h` to the project's `AppName-Prefix.pch` file. This will then

allow the developer to make calls to the SUMClient framework within any class file in the project.

4.3.2 Authentication

Before an application can utilise the SUM framework's features, it must first authenticate with the SUMServer to ensure that access has been granted. After registering with the SUM framework developers would be provided with their unique application authentication key to be added in the application launch method (Figure 4.10). This key is used to verify the application with each launch to ensure access is granted. In the event of no active Internet connection, the application will proceed to operate as normal using the local data without access to any SUMServer features i.e. model requests, or data aggregation.

Upon the first launch of the SUM enabled application, if no user credentials are identified, the framework will automatically prompt the user to login with his/her username and password. Successful login details are then stored for future sessions and can be removed via the application settings panel within the SUM configurations. Once logged in, SUM will activate the logging procedures and begin capturing user interactions.

```
- (void)applicationDidFinishLaunching:(UIApplication *)application {
        ///Create an instance of the SUMClient core
        sum = [SUMCore sharedInstance];
        ///Set the application appkey + authenticate with the SUMServer
        [sum setApp:@"A5MToxNC45MTk5MDIa212OS0yOCAwNjo0Nac3a03ea08b199ae0jAxMi0wb9d1d070b08_MTM3Mz"];


        /// rest of application launch code goes here ///

}
```

Figure 4.10 Code required for SUMClient initialisation and application authentication

4.3.3 Interface Adaptation

The SUM framework (version 1) is not an autonomous interface generating system like the SUPPLE (Gajos et al., 2007) system, which creates interfaces optimised for user abilities and removes the interface designer's control and input. Instead, SUM has been designed to supply an application with the optimal parameters for a user's current needs and abilities; allowing developers to receive this information and decide on the appropriate actions to respond with. For example, the WalkType system (Goel et al., 2012)) contains multiple interface layouts: one for small targets with more on the screen and another with larger targets spanning the full width of the screen, with fewer items visible at once. Using SUM, the system could have requested the optimal target size for the user's current abilities then selected the appropriate interface that supported those requirements.

4.4 Conclusions

The design and development of SUM Framework were presented in this chapter. SUM is a user modelling framework designed to collect measurements of touch performance and adapt interfaces to match each user's abilities. Background and related work to support the need for the development of SUM were presented in this chapter, along with some refinement of the scope and direction of this current research. The core concepts of SUM were outlined and the technical design and architecture of the framework were described. This was followed by a discussion of the intended workflow of building applications using the SUM framework.

Chapter 5. Laboratory Evaluation with SUM

This chapter presents the laboratory user evaluations conducted with 12 participants having visual and motor impairments. The evaluation uses three applications in which the SUM framework is embedded (Chapter 4). The primary objective of this study was to evaluate the application of the SUM framework as a method for capturing user measurements and providing individual interface adaptations to touchscreen devices. Secondly, this study aimed to further explore the characteristics and behaviours of users when interacting with mobile touchscreen devices by capturing detailed measurements of their onscreen interactions. The user study details the process of embedding the SUM framework into touchscreen applications, and outlines the analysis techniques used to retrieve accurate measurements of user performance from SUM interactions.

5.1 User Study

This section presents the laboratory study carried out with 12 individuals with various levels of visual and motor abilities. The objectives of this investigation were: firstly, to evaluate the application of the SUM framework as a method of measuring individual needs and abilities; secondly, the study explored adaptation techniques designed to improve touchscreen accessibility and better support the users' individual abilities and needs. Finally, this study aimed to review the use of shared user models between applications to support accurate modelling of user abilities. To understand the effects of the SUM adaptions, the following two-interface conditions were defined:

Static: The static interface condition represents the default interface designs and interaction behaviours. Participants within the static interface group would receive applications using the manufacturers default touch gesture recognisers, and interface elements used the guideline (Apple, 2009) target properties (i.e. font sizes and target bounds).

Adaptive: The adaptive interface condition represents the interfaces made possible by the SUM framework. Participants within the adaptive interface group would receive applications using the SUM adapted touch gesture recognisers (i.e. personalised touch durations and offsets), and interface elements (i.e. target bounds would be adjusted to the individual, and interface modalities would adjust based on preferences). These adaptions were performed to the Indoor navigation and TV Guide applications before the participants used them, and adaptions occurred continuously during the Target Practice tasks.

5.1.1 Participants

As reviewed in previous chapters, persons with low vision and motor impairments experience challenges when interacting with mobile touchscreen technologies. The preliminary user evaluation involved older adults who exhibited characteristics of low levels of vision and motor control, helping to expose the challenges and identify strategies to mitigate these barriers. This laboratory evaluation aimed to examine the utility of shared user modelling for individuals with visual and motor impairments, providing adaptations that matched their individual abilities and not the stereotypical characteristics of disabilities. Furthermore, the research intended to investigate the feasibility of a model that would allow interaction features to be tailored to an

individual. For these reasons, users with a variety of abilities were recruited to evaluate the performance of the models across a diverse range of users. A total of three male and nine female adults were recruited. They ranged in age from 21-71 (M=54, SD=20) and all possessed characteristics that would qualify them as low vision and/or motor impaired. To ensure this evaluation exposed challenges pertaining to the physical interactions of mobile touchscreens and not the barriers relating to using digital technologies, all participants were required to own and use a mobile phone (although not necessarily a smart phone) and have a computer. Table 5.1 provides information about the participating individuals and their characteristics. See appendix 2 for information sheet and consent forms.

ID	Age	Gender	Method	Touchscreen Experience	Group	Impairment	Current Accommodations
P1	67	Female	Static	None	VM	Tremors in hands, short-sighted	Medication to suppress symptoms
P2	58	Male	Static	Self-service machines	VM	Spinal injury, muscle spasms, sensitive to light	Medication to suppress symptoms, powered wheelchair.
P3	57	Female	Static	None	M	Dopa-responsive dystonia, muscle cramps, tremors in hands	Reduced sensitivity of keyboard and mouse to minimise errors.
P4	66	Male	Static	Self-service machines	M	Spinal injury, hand and wrist pains	
P5	66	Female	Adaptive	None	V	Retinal detachment, macular degeneration, diplopia	Guide dog, magnifying glasses, screen reader software on PC
P6	65	Female	Adaptive	None	VM	Macular degeneration in left eye, tremors in hands	Powered wheelchair, full-time carers, mobile with large buttons
P7	67	Female	Adaptive	None	V	No binocular vision, reduced vision in left eye.	Magnifying glasses
P8	21	Female	Adaptive	Has a touchscreen phone (single touch)	M	Hypermobility syndrome, locking joints and tremors in hands	Wheelchair, medication to suppress symptoms
P9	71	Female	Adaptive	None	M	Myalgic encephalomyelitis, muscle twitches and spasms in arms and hands	Medication to suppress mobility symptoms not cognitive
P10	64	Female	Static	None	V	Reduced vision in left eye	
P11	23	Male	Adaptive	Has used ipod touch before, self-service machines	V	Registered blind, issues adjusting to changes in light levels	Monocular, screen magnification on PC, mobile with large buttons
P12	22	Female	Static	None	VM	Ataxia with oculomotor apraxia, reduced levels of vision, muscle twitches in hands and difficulties with fine motor control	Powered wheelchair, full-time carers, mobile with large buttons

Table 5.1 Overview of participant information

5.1.2 Apparatus

The mobile device selected for this study was the second generation Apple iPod touch (Figure 3.1) running iOS 3.0 as used within the preliminary research (Chapter 3). While this version of the OS shipped with a suite of accessibility features including VoiceOver text to speech, these features were not available to the second generation devices used within this study.

5.1.2.1 Experimental Apps

The three iPod Touch apps specifically developed for this study were: Target Practice, Indoor Navigation and a TV Guide. Each application required the participant to interact with on-screen controls using a single touch, known as a *tap* gesture. Within the Target Practice and Indoor Navigation applications the device responded only to a tap gesture. Whilst the primary input for the TV Guide was a single tap, it also required the users to swipe vertically to scroll through the TV listings. The interface adaptations created in the apps for visually impaired users also provided for scaling of text sizes and provided a text-to-speech option. Those adaptations, however, were largely based on user preferences as input, rather than abilities input. For the purpose of the current SUM Framework investigation, these adaptations will not be analysed in detail, although the user satisfaction ratings (to be described later) will be reflective of the fact that adaptations were made for the visually impaired participants.

Each of the three experimental apps was designed and built to conform with the iOS interface guidelines (Apple, 2009). For all three, interface elements were given minimum bounds of 10mm (60 pixels on this device) identified in previous research

to be the optimal target size for daily users of these devices (S. Lee & Zhai, 2009; Parhi et al., 2006; Y. S. Park et al., 2008). The SUMClient framework was embedded into all the apps for device monitoring and communication with the SUMServer for user modelling.

The apps shared user models by synchronising with the SUMServer after each application task was complete. Similarly, at the beginning of each application task the researcher would request the latest user model from the SUMServer for the participant. Adaptations were applied specifically to users receiving the adaptive interface condition (Table 5.1) before the participant used the application; default values were employed for the static interfaces. However, all participants used the Target Practice application in both interface conditions. Furthermore, the Target Practice application performed interface adaptations in real-time (i.e. the targets would scale up and down, likewise the minimum and maximum tap durations would adjust). Complete details of the adaptations are outline in section 5.1.3.1.

5.1.2.2 Target Practice

The Target Practice app (Figure 5.1), was designed to capture baseline data about participants' abilities at the time of test. As users with disabilities can often experience fluctuations in ability, this baseline data was essential. The app generated 200 pairs of targets within the screen. Users were asked to tap the 'green targets only' (none of the participants indicated that they were colour-blind). If the participant does not tap the target within 10 seconds of it appearing on the screen then the application moves onto the next target pair, and generates two new targets. Likewise if the participant touches the wrong target, or an empty location on the

screen, the application will accept the input and move onto the next target pair until the game is complete. The target positions were pseudo-random as constraints were applied to the position generator to ensure good distribution of the targets. Specifically, the screen was divided into three sections vertically and horizontally, with the centre section twice the size of the other two for the vertical divisions (1:2:1) and three times the size of the other two for the horizontal divisions (1:3:1).

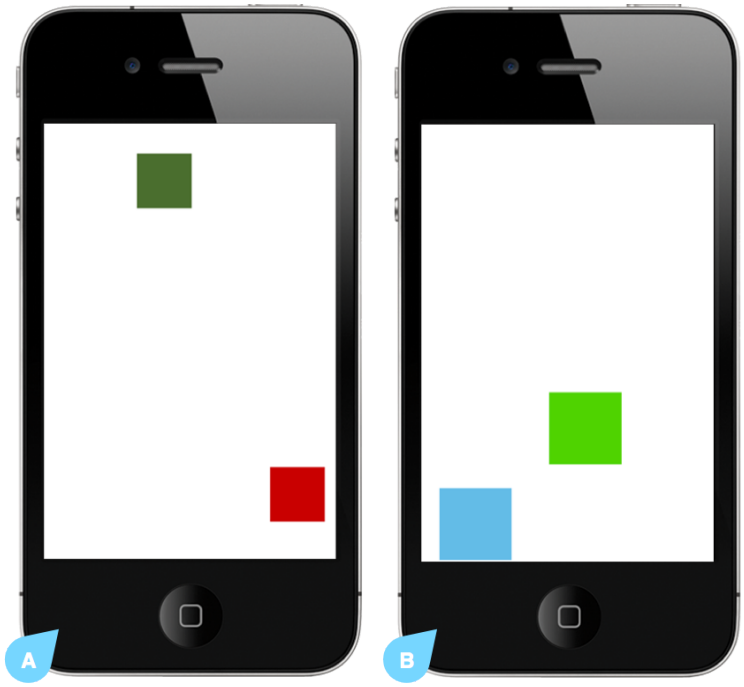

Figure 5.1 Target Practice gameplay screens: static interface (A) and adaptive interface (B) conditions.

5.1.2.3 Indoor Navigation

In the Indoor Navigation app shown in Figure 5.2, the user interactions and appearance of this application remained unchanged from the version used with the exploratory evaluations (Chapter 3). However, the technical design of the application was refined based on the limitations identified from the previous evaluation. The updated application preloaded all instruction content and media elements prior to the

user undergoing the way-finding task. This change ensured that all instructions were successfully loaded and maintained a consistent user experience regardless of WiFi network strength or connectivity. The indoor navigation application was designed to provide users with individually tailored way-finding instructions; it made adaptations to the interface delivered to the user such as scaling visual interface elements. Way-finding routes were kept consistent for each participant.

Participants were asked to complete two indoor way-finding tasks within the University's School of Computing building (an unfamiliar environment for them), using only the instructions provided by the Indoor Navigation app. This navigation was accomplished via stored location information and the user's ability to match descriptions or images to their current location (Montague, 2010). The navigation tasks required that the participant physically navigate from one location to another. Each participant performed both routes, however the order in which the participants carried out the tasks was counterbalanced.

Instructions were provided as text way-finding directions and accompanying images, with the option to have text read aloud using the audio button.

Figure 5.2 Indoor Navigation application instruction screen, static interface (A) and example of a low vision interface of the adaptive interface (B)

5.1.2.4 TV Guide

The TV Guide app (Figure 5.3), provided users with fixed TV listings for seven channels and 28 programmes. Users were asked to find specific TV programmes. To do this, they needed to browse through lists and grids of channels and programmes. Upon finding the programme they were asked to name the TV channel it airs on; transmission date and time; read aloud the description text and then state any available access formats (such as Audio Description being available).

Figure 5.3 TV Guide application, programme list view for static condition (A) and example of a low vision interface of the adaptive condition (B); programme details view for the static condition (C).

5.1.3 Procedure

The evaluations were split between two sessions to ensure participants would not experience exhaustion or fatigue during the study. Each session consisted of testing two of the experimental apps, administration of a paper and pencil questionnaire and a informal discussion with the researcher. The total session time for this first session was between 45 and 70 minutes, with most participants completing in one hour. They were gifted vouchers (worth £10) for their participation in each session. The tasks were structured in two sessions, as follows:

5.1.3.1 Session 1

Experimental sessions were conducted in a university environment and began with participants being given an overview of the research and each of them reading and

signing the Informed Consent form prior to the start of the study. All participants then completed the following steps, in order:

Interview: This background interview collected data about participants' mobile phone use, computer experience, handedness, and experience with touchscreen devices. Of the 12 participants, all of them used a mobile phone on a daily basis. Only three owned touch-enabled smart phones and they were the only participants to have previously used a touchscreen mobile device. The researcher also discussed participants' abilities, asking questions about the use of glasses and assistive technology devices.

Target practice tasks: Each participant did the target practice task twice – once with the Adaptive interface and once with the Static interface. Participants' touches were analysed throughout the target practice task to produce the interaction changes for the Adaptive condition. These changes were uniform scaling of target sizes and adjustments to touch duration bounds. Target scaling factors were calculated based on a participant's offset distance (x) from target centroid when tapping targets.

$$k = \frac{\bar{x}}{2}$$

Equation 5.1 Target scaling factor, where *k* is the scaling factor and *x*-bar is the mean offset distance from the target centroid.

To eliminate outlier values, the minimum (10th percentile for individual user) and maximum (90th percentile per individual users) target offsets were removed (Chapter 4).

Static interfaces had consistent target bounds of 60x60 pixels and no minimum or maximum touch duration. The order of these two conditions was counterbalanced

between participants such that half received the Static interface first. While using the target practice app, participants were asked to relax in an armchair (or their own wheelchair). Each test with the target practice task lasted about four to five minutes (Static, M = 291 seconds, SD = 278.2 and Adaptive, M = 213 seconds, SD = 111.5).

Setting Preferences: Following the target practice task, the user inputs were synchronised with the SUMServer. In addition, participants were asked about their preferences in terms of audio and text presentation and their preferred volume level was set. These preferences were entered into the user model by the researcher.

Indoor Navigation app: Participants were then given the Indoor Navigation app with the interface condition matching the method for their allocated group. The interface method allocation of each participant is shown in Table 5.1.

For the Adaptive method, two types of changes were made. The first applied the scaling factor and touch duration bounds from the target practice task to the Indoor Navigation interface elements. In addition, the individual's preferences for text, audio and images were applied, thus altering the modalities present in the interface. For the Static method, the interface was shown with no accessibility adaptations, as with the target practice app. Participants averaged about four minutes to complete each of the two Indoor Navigation tasks. Participants were able to complete all of the tasks with one exception in the Indoor Navigation study. This inability to complete one task was due to a technical disruption caused by a loss of Wi-Fi connectivity.

Questionnaire for Indoor Navigation app: The Simple Usability Scale (SUS) questionnaire (Brooke, 1996) was administered. This questionnaire consists of 10

questions about usability, with participants being asked to respond on a five point Likert scale. Although this questionnaire was initially designed to be a paper and pencil test, it became clear with the first few participants that it was difficult for them to read even the 14- point text and/or mark their answers. The researcher therefore adopted the procedure of reading the statements aloud to participants and asking them to verbally indicate their responses ("Strongly agree" "Agree" "Neutral" "Disagree" or "Strongly disagree").

Informal Discussion: Each session ended with the experimenter asking the participant for feedback on her/his experience. This was augmented with the researcher's recorded observations.

5.1.3.2 Session 2

The structure of this second session was similar to that of session one. The total duration of this second session was about one hour, ranging between 50 and 90 minutes for the 12 participants. The time between the two sessions varied with the second session taking place from one week to three months after the first session, depending on participant availability.

Interview: The experimenter began by asking participants about any known changes in their abilities since the previous session. This proved to be useful, particularly in one case, in which the participant had had a change of medication and was more comfortable with the touchscreen than in the first session. The experimenter asked questions about their TV viewing habits, whether they used subtitles (captioning),

on-screen TV guides, on-demand services, recording systems, and how they planned their TV viewing.

Target practice tasks: Participants repeated both target practice tasks, using the Static and Adaptive interfaces in the same order they had used them in Session 1.

Vision test: Participants were given a Snellen eye test for both distance and reading. The results of this reading test were added to the individual's user model to identify a font size for optimal viewing. iOS uses a variation of Helvetica by default, and its pixel size is 16px for normal text. The assumption was that this text would be the minimum size anyone should be given. Thus, 20/20 vision was allocated 16px, the rest of the font sizes were calculated based on this value; for example, for 20/50 vision, 50/20*16px (our default size for this font) therefore the size was set at 40px.

TV Guide app: Each participant then performed the tasks for the TV Guide app. Participants were tested with either the Static or Adaptive interface depending on their assigned Method as shown in Table 5.1. For the Adaptive interface, elements were adjusted to be consistent with the methods used in the Indoor Navigation version, with the addition of the Snellen results being used to scale the text size. The static interface again had no adaptations and participants received default text sizes and touch properties for the device. The tasks for the TV Guide app took, on average, 20 minutes to complete.

Questionnaire for TV Guide app: Participants were verbally asked the questions from the SUS questionnaire, with respect to their experience with this second sessions' experimental app.

Informal Discussion: Participants were given the opportunity to comment on any features of the apps they wished to. The experimenter also followed up on any observations made during each of the participants' sessions (Figure 5.4).

Figure 5.4 Informal discussion and a participant sharing her experience of the laboratory study and using the applications on the touchscreen device.

5.2 Results

The objectives of this study were to evaluate the use of the SUM framework to both accurately measure user performance and provide appropriate interface adaptations. Interface adaptations were made based on the user's previous interaction data with the experimental applications, for example, using the interactions within the target practice game to perform adaptations to the interface of the Indoor Navigation application. This approach allowed the interfaces to be tailored to the user's abilities and needs before he/she ever used it (these adaptations excluded the first application,

since no prior data existed). The study also aimed to further explore the characteristics and behaviours of touchscreen interactions, building on the investigation within the preliminary research (Chapter 3).

5.2.1 Qualitative

Qualitative measures were captured through researcher observations and informal discussions after participants completed the application tasks.

5.2.1.1 Applications vs. Tasks

Previous works relied on explicit calibration exercises to obtain accurate measurements of users' performance (Gajos et al., 2007; Trewin et al., 2006), the goal of SUM was to remove the need for ability elicitation tasks by capturing measurements of user interactions in the background of real applications. Therefore, this study involved participants using applications that would either be found already in the mobile marketplaces (Target practice and TV Guide) or served a real world purpose (Indoor Navigation).

Whilst SUM aims to address the challenges of background user modelling, it was also important that the participants believed the applications served a purpose beyond measuring their performance. *Did the participant believe they were using a real world application, or did it still feel like a calibration task?* Participants were asked to discuss their experience of using each application, and comment on their desire to use such an application again outwith the evaluation. All participants agreed that the TV Guide application would be very useful, and it was easier to find shows than using the printed Radio Times TV guide. Likewise, most participants (excluding *P11*) believed the Indoor Navigation application would be extremely

useful in their lives. *P5* commented "*It would be great for the visits to Ninewells* [hospital] *for my checkups and appointments, sometimes it's like a maze"*. However, none of the participants particularly enjoyed using the Target Practice game, *"It seemed very tedious and boring…I would not play this at home"* expressed *P8*. Of the three applications, Target Practice most resembled a calibration task. Participants were guided to select an onscreen target with no rationale for why they must perform this task, nor an incentive to continue to interact with the application.

Participants were asked to suggest possible uses or applications that would be beneficial to have on such a mobile touchscreen device. Again, all participants suggested the TV Guide application would be desirable, but with the additional functionality to set reminders for TV Shows or remotely record them. Other suggestions included accessing emails, video calling with family and general web browsing, medication and appointment reminders, playing games and using social networks. All of the ideas are currently possible and exist as applications on these touchscreen devices. Furthermore, the interaction methods of the suggested applications closely resemble those of the applications within this user study. Thus, future evaluations should seek to incorporate these more desirable and appealing applications to gain self-motivated interactions, as opposed to forcing participants to engage with content that they have little to no interest in.

In conclusion, two of the experimental applications successfully resembled real world applications, and convinced the participants that they were not simply performing an ability elicitation task. Nevertheless, all participants felt that the

Target Practice game was more of a task than an enjoyable application, suggesting that more engaging applications would have been better.

5.2.2 Quantitative

With regards to touch errors, it was hypothesised that the SUM adaptive interfaces would result in fewer errors than the static interface. To test this, the touches within the Target Practice app were examined. For this task, it was clear if an error was made. If the participant hit the wrong target or touched outside of the target this was counted as an error.

Overall, there were 3,997 touches (3,603 within Target Practice app) with the Static interface for the three apps, and 3,259 (2,989 within Target Practice app) for the adaptive interfaces.

In the target practice task, participants were asked to touch all the green targets. Even though there was a green target in each pair of target stimuli, in some instances participants made no attempt to tap the target. This was due to the fact that some believed the dark green target to be a shade of brown or black, not green. Therefore, touch error rate was based only on the attempted targets within this task.

As hypothesised, the SUM Adaptive interfaces produced fewer touch errors than did the Static interface $t(11) = 1.977$, $p < .05$, one tailed ($d = .632$). The mean number of touch errors per target practice interface was 18.83 (SD=19.41) and 27.67 (SD=26.36) for the Adaptive and Static interfaces. The touch data for target practice taps was segmented into the 1:2:1, 1:3:1 sections used to distribute the touch targets to investigate the spread of the errors. Figure 5.5 presents the error rate heat maps for

the Adaptive and Static interface conditions. Whilst the Adaptive interface has lower error rates, both interfaces have a similar pattern with lower error rates in the bottom (vertically) and right-hand side (horizontally). The researcher had expected to see smaller numbers of errors in the bottom of the screen as the distance from the arm support increases (Guerreiro, Nicolau, Jorge, & Gonçalves, 2010a). Although our participants were not asked to hold the device in a particular manner, the researcher observed similarities between their hold and touch configurations. For example, participants grasped the device in the left hand and used one finger or a combination of thumb, index and middle fingers from their right hand. As the device is thus positioned closer to the origin of the participant's right hand (used for interactions), the distance to the target is lower in these areas. This could explain the lower error rates along the right-hand side.

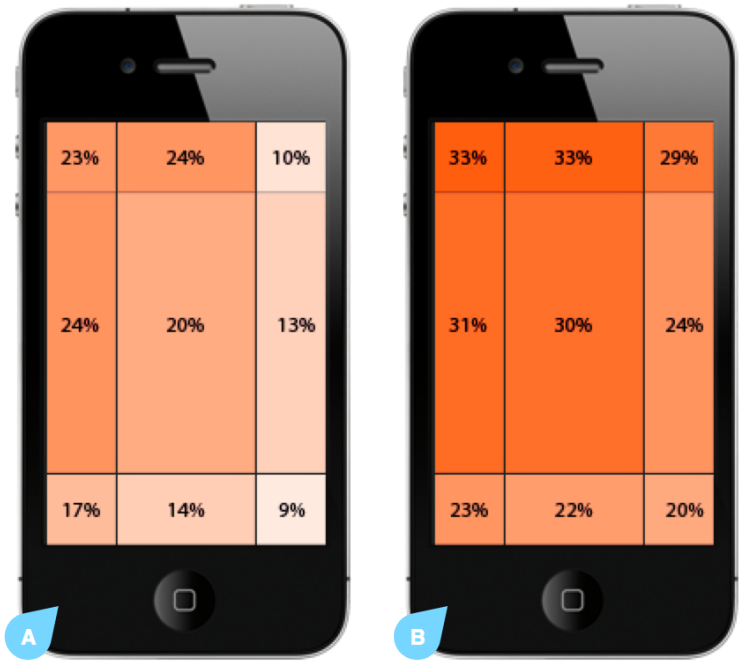

Figure 5.5 Error rates within the Target Practice game for each of the screen location segments, (A) adaptive interface (B) static interface condition.

As well as identifying error rates within specific screen locations, our data also revealed the touch locations relative to the centre of the targets. Tables 5.2 and 5.3 summarise the finger locations relative to all targets hit or missed during the target practice exercise. The term origin refers to hitting the target's centroid x or y coordinate exactly (dependent on the screen locations, vertical or horizontal). Of these, statistical analyses showed significant differences in the error locations for the vertical-vertical errors (Table 5.2), $\chi^2(4) = 29.84$, $p<.001$. Specifically, within the vertical locations there was a relatively even distribution between participant touches above and below the origin, but for the horizontal touch locations the participants selected the right of targets the majority of the time.

Vertical Screen Location	Vertical Offset Location		
	Above	Origin	Below
Top	45.6%	3.0%	51.5%
Centre	39.4%	2.7%	57.9%
Bottom	46.3%	2.0%	51.7%

Table 5.2 Summary of vertical touch locations relative to the target centroid within vertical screen locations.

Horizontal Screen Location	Horizontal Offset Location		
	Left	Origin	Right
Left	30.2%	1.5%	68.3%
Centre	29.5%	2.2%	68.3%
Right	31.9%	1.3%	66.8%

Table 5.3 Summary of horizontal touch locations relative to the target centroid within horizontal screen locations.

The capacitive touchscreens found in the iPod devices are highly sensitive and able to detect touch input with next to zero finger pressure, often seen as one of the advantages of the technology. For some individuals, however, this highly sensitive

screen is challenging. Three of our participants experience intermittent hand tremors, and consequently found themselves making unintentional taps. P1 and P9 both own iOS devices. However, P9 chooses not to use her iPod touch because of these issues. Instead she uses an LG touchscreen phone because *"it has a much lower sensitivity than the iPod"*.

Figure 5.6 illustrates the tap durations of each user collected from interactions within the mobile apps. Tap durations were statistically significantly different between the participants, Welch's $F(11, 1245.43)=905.5$, $p<.001$. Post-hoc analysis revealed significant differences at the level ($p<.00064$ using Bonferroni correction), between each participant pair with the exception of P1 (.044) and P8 (.038) ($p=.357$); P3 (.177) and P11 ($p=.948$); P5 (.335) and P6 (.415) ($p=.075$); P5 and P12 (.264) ($p=.096$). These results suggest that there could not be a minimum and maximum duration that would be optimal for everyone. The SUM Framework used the duration data and adjusted the level of sensitivity by applying more restrictions to the timings of the tap gesture recogniser. In an attempt to reduce the number of involuntary inputs, the tap recogniser was given minimum and maximum durations for the Adaptive interfaces. The default tap gesture recognisers to not use these parameters. Using the SUM gesture recognisers P9 was able to notice the benefits during the second session when she tested the recogniser with a tap preceded with an echo of the tap (approximate to the common interaction challenge she faces as a result of her hand tremors). However, the SUM tap gesture recogniser was able to identify the original tap, and disregard the second tap completely.

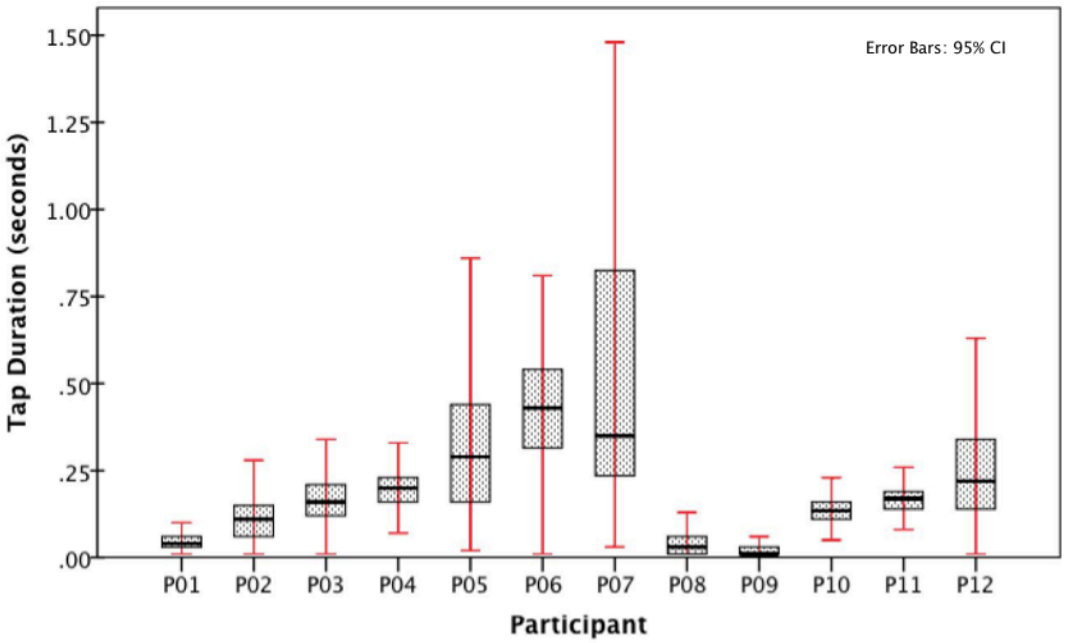

Figure 5.6 Durations of tap gestures for participants across both study sessions.

Both the Target Practice and TV Guide apps required only a single tap input to interact with the interface elements. The TV Guide app makes use of scrollable panels to present more content within a single page, such as a list of all programmes. These scrollable panels caused a great deal of confusion for P12 when trying to locate TV programmes positioned further down the grid off-screen. Since she had never used a smart phone until taking part in this study, her working knowledge of grids and lists came from her computer experience. To look for content not on-screen, P12 looked for scroll bars as well as previous and next page buttons. When asked by the researcher if she was able to find *The Inbetweeners* TV programme within the page, P12 stated that *'it wasn't on the page'*. The researcher then prompted the participant that there was more content below, and she was then able to perform the required scroll gesture to complete the tasks.

The iPod touch is capable of two types of scrolling. One, free scroll with acceleration, will move faster or slower depending on the swipe input speed and will keep scrolling until it decelerates and stops. The second type of scrolling uses a paging effect so that regardless of the swipe input speed, the panel will only scroll one page. Both types of scrolling were incorporated in the TV Guide app: free scroll for the 'all programmes grid' and page scroll for the 'all channels' grid and 'programme detail' pages. P5, P6 and P12 shared similar problems when using the free scroll. Their comments included *"When you do it [scroll], it just keeps going and I can't read it," "I can't see it quick enough"* and *"I don't like it moving past"*.

The researcher also observed changes in grip style when participants were required to scroll rather than tap. When scrolling, the device was repositioned and given a firmer grip to ensure that it wasn't dropped when performing the necessary swipe gesture. For some participants this resulted in unusual behaviour of the device caused by their unintentional touches when tightening their grip. While the iPods have a bezel edge on all four sides of the screen, this was much too small for a number of our participants to hold without creating involuntary touch input. A small bezel appears to be a design trend for touchscreen technologies, as it maximises screen size but minimises device dimensions. This trend may make the devices more challenging for users with diverse needs.

The SUS questionnaire is designed to produce measures of usability. We hypothesised more positive usability ratings for the Adaptive interfaces than for the Static interface. Recall that in the Adaptive condition the interfaces were designed for meeting individual needs for touch, visual display and text-to-speech preferences.

In total, there were 12 SUS scores for the Adaptive interfaces (six for both the Indoor Navigation and TV Guide app), and 12 SUS scores for the Static interfaces, but each participant rated different apps in the adaptive and static conditions. There was only a small and statistically non-significant ($p>.05$) difference in the SUS scores in these two conditions (the mean usability rating for the Adaptive interfaces was 3.33 (SD=0.27), compared with the mean usability rating of 3.10 (SD=0.71) for Static interfaces). There were likely various reasons contributing to the small difference and the fact that users, overall, did not give high ratings to the adaptive interfaces. Primarily, participants' use of the apps was limited. Not only did participants have limited time with the apps tested, it is important to note that they only viewed the apps in one experimental condition. Thus, they did not directly compare the apps under both the Adaptive and Static conditions, the apps were new to the participants and, regardless of testing condition, they had to learn how to use the new app. Consequently, their comfort with the apps, regardless of interface, was likely to be limited. Establishing more extended testing with the apps and in multiple interface conditions should help to better understand the extent to which the SUM Framework adaptations are perceived.

5.3 Conclusion

This chapter reported on a laboratory-based user study that investigated mobile touchscreen interactions. The evaluation explored the use of the SUM framework as a method of collecting natural application interactions for the user modelling of individual abilities, and for providing interface adaptations between distinct applications, in order to improve the accessibility and usability of the devices. Two

interface conditions were evaluated, static - representing the existing design guidelines and best practises; and adaptive – applying interface adaptations based on the individual's preferences and interaction abilities. The results of this user study not only demonstrated that participants produced fewer target selection errors when using the personalised (adaptive) interface, but also showed that the timing interaction behaviours were statistically significantly different between participants. Interface adaptations were provided by leveraging individual's previous interactions (within other applications) and their interface preferences, to adapt layouts and interaction parameters before they used the application. These results support the proposed approach to measure individual's abilities from application interactions, and build user models that can be shared between applications for interface adaptations to improve access and usability

Chapter 6. Revising the SUM Framework

This chapter details the revisions made to the design and implementation of the SUM framework in light of the laboratory user evaluations (Chapter 5). The chapter begins by outlining the technical limitations associated with the earlier version of the SUM framework in relation to the practical evaluation. Maintaining a consistent structure to the earlier technical outline of SUM (Chapter 4), this chapter then describes the individual revisions carried out to address prior limitations and extend the capabilities of the framework. The chapter covers the modifications to the overall structure and internal changes to both the SUMClient and SUMServer. Finally, the chapter demonstrates the revised methods of embedding the SUM framework into third party applications and the adaptation methods applied to create interactions tailored to individual user's needs and abilities.

6.1 Earlier Limitations

The laboratory user evaluations with SUM (Chapter 5) highlighted a number of limitations of the SUM framework as a solution for modelling individuals with variable abilities. While the previous discussion of these limitations was in relation to the controlled laboratory environment with limited windows for data collection and the large periods of time between user interaction sessions, this section will discuss the technical limitations of the SUM framework version one and the constraints they imposed on the evaluation.

Measurement accuracy: The largest limitations of SUM framework version one stemmed from the approach to data collection and local storage. In the interest of

reducing the local storage requirements and minimising network usage, SUMClient would process the raw sensor values from touchscreen interactions and only store the resulting parameters of the overall gesture. For instance, the *total duration* would be stored instead of the *individual* durations between the *touch begin*, *touch move* and *touch end* states of the touch gesture. As a result of this design choice, the possible features to describe the touch interactions were limited to a gestural level i.e. measurements between taps rather than understanding the user behaviours between *touch begin* and *touch move* states. As a result it was not possible to investigate the sub-gesture movements and accelerations.

Performance: directly related to the aforementioned touch sensor processing, the SUM framework injected additional method calls within the process pipeline of the touch sensing. The added steps to process the raw sensor data into a complete touch gesture and store it periodically caused the application interfaces to lock with the result that the participants in the evaluation study experienced a lagging effect between their touches and the resulting actions.

Similarly, the SUM framework was constantly accessing the device sensors and network connection, which was a massive drain on the device's battery power supply. This behaviour had little effect on the hour-long user evaluation within the laboratory, where the researcher could connect and charge the device at the end of each session. If the same framework were to be used within a study in the wild, then participants would need to regularly charge the devices throughout the day as a result of SUM additional drain on the battery. This would clearly be unacceptable from a user standpoint.

6.2 Revisions to SUM

This section details the technical changes made the SUM framework based on the outcomes of the laboratory user evaluation. These changes included revisions to the overall structure of the framework, altering the core functionality of both the SUMClient and SUMServer. The goal of these revisions is to enable the SUM Framework to support longitudinal evaluations in the wild.

6.2.1 Structure

While the high level architecture of SUM was not changed from what was described previously (Chapter 4), the roles and functions of the SUMClient and SUMServer were adjusted to improve both performance and accuracy of SUM. The user feedback and observations collected within the laboratory user evaluation with SUM (Chapter 5) identified interface response lag as a result of the additional function calls of SUM within the touch event pipeline. Similarly, pre-processing the touch gestures on the device before storing them resulted in a loss of detail for finer grain analysis of the touch gesture features. To address these limitations and extend the SUM framework's capabilities a number of developmental changes were made to the core structure of SUM.

6.2.2 SUMClient

Figure 6.1 details the updated SUMClient's internal architecture of the core components. There are three fundamental revisions between this version and the previous:

1. Sensor measurements are no longer coupled with user interface components.
2. Local data storage has been extended to support longer usage by automatically clustering user interactions into sessions.
3. Data synchronisation and management has been overhauled to automate the process in support of remote user evaluations.

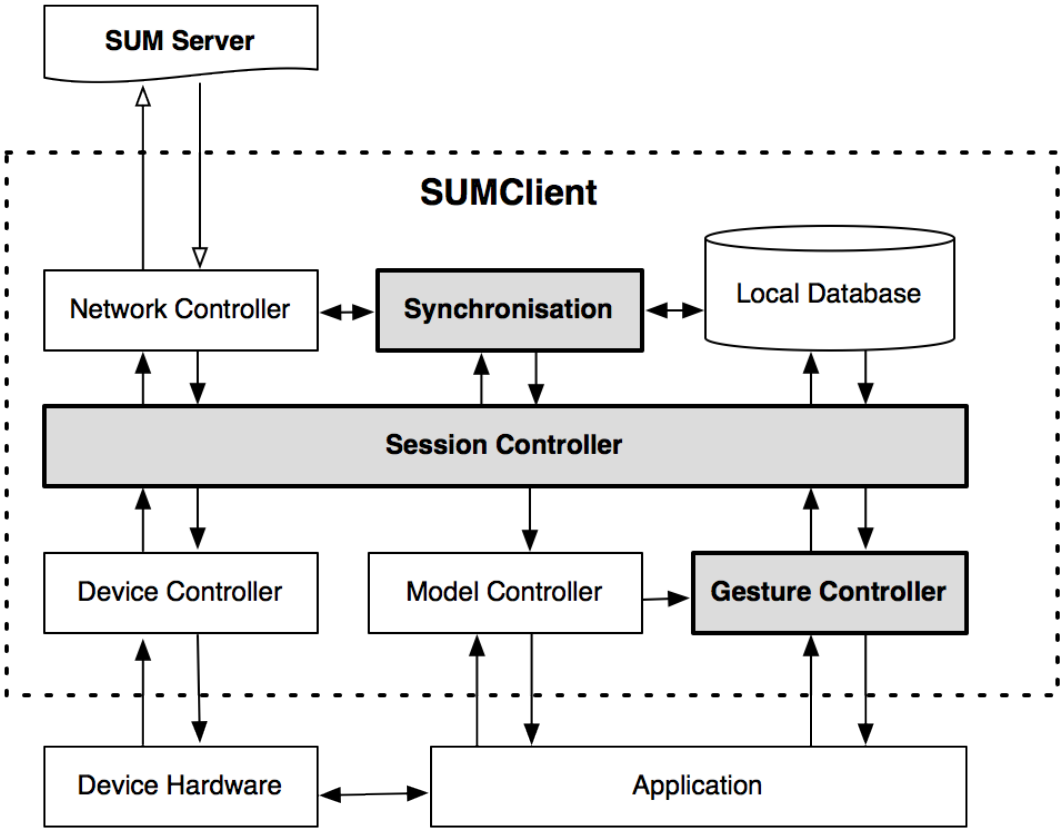

Figure 6.1 SUMClient revised version software architecture showing the communication between internal components and third party applications.

Touch Sensing: previously, the SUMClient provided applications with overwritten user interface components to be substituted for the device's default interface components. These SUM interface components accessed user interactions via the *touch begin, touch move* and *touch end* events within each interface component. This technique required developers to change every instance of UIControls within their application interface in order to allow SUM to capture the user's touch events throughout the application. Since the development of the original SUM framework the iOS (and Android) operating system have introduced *GestureRecognisers,* altering the methods of creating interface controls that respond to user interaction. Developers can easily create their own customised user interface components, then define a GestureRecogniser object to handle user interactions, including any touch and motion gestures. This new approach allows developers to define a single GestureRecogniser object, and use it to handle the same type of gesture for completely different interface components. Leveraging the GestureRecogniser functionality, SUMClient now contains an overwritten GestureRecogniser class with all the necessary measurement code. Furthermore the GestureRecognisers provide straightforward mechanisms for adjusting the parameters that define a successful interaction which is beneficial for interface adaptations (Section 6.3).

Motion Sensing: SUMClient's Device Controller captures measurements of device motion, using the available motion sensors, i.e. Accelerometer or Gyroscope. Sensor fusion (J. Lee & Ha, 1999) methods, combining the sensor readings from accelerometers and gyroscopes to produce more accurate measurements of motion, have also been included within the device operating systems since the original SUM framework. SUMClient was updated to use these new techniques and so capture

device motion with higher accuracy. The process for storing this data has remained the same. However, where the earlier version of the SUMClient would activate device logging and capture motion events at 100Hz with no consideration for the impact on battery life, or the actual motion happening; the framework has been updated to observe the sensor readings. If the device motion falls below the threshold value for 10 seconds then the framework stops capturing the values and reduces the sensor rate to 2Hz. Once the device motion crosses the threshold again the rate is increased and the framework continues to capture the data, as used in (Pham, Plötz, & Olivier, 2010). By reducing the sensor refresh rate during inactive – motion periods, SUM can lower the performance footprint and effects on battery life. Similarly, through stopping the logging of motion data during these periods and only storing the motion data that represents active device motion SUM is able to reduce the amount of data storage required. Both are important factors when considering in-situation deployment within longitudinal evaluations.

6.2.2.1 Longer Usage and Data Storage

In relation to the changes made in the data collection methods, the storage structure has been revised to reflect the distinction and decoupling of application interface components from user gesture interactions and storing the raw unprocessed values from the various sensors. The other major change to the structure of SUMClient's data storage is the newly included *Session* object, which relates to the need for longer collection periods to understand long-term usage behaviours. Figure 6.2 presents the revised table structure of SUMClient's local storage.

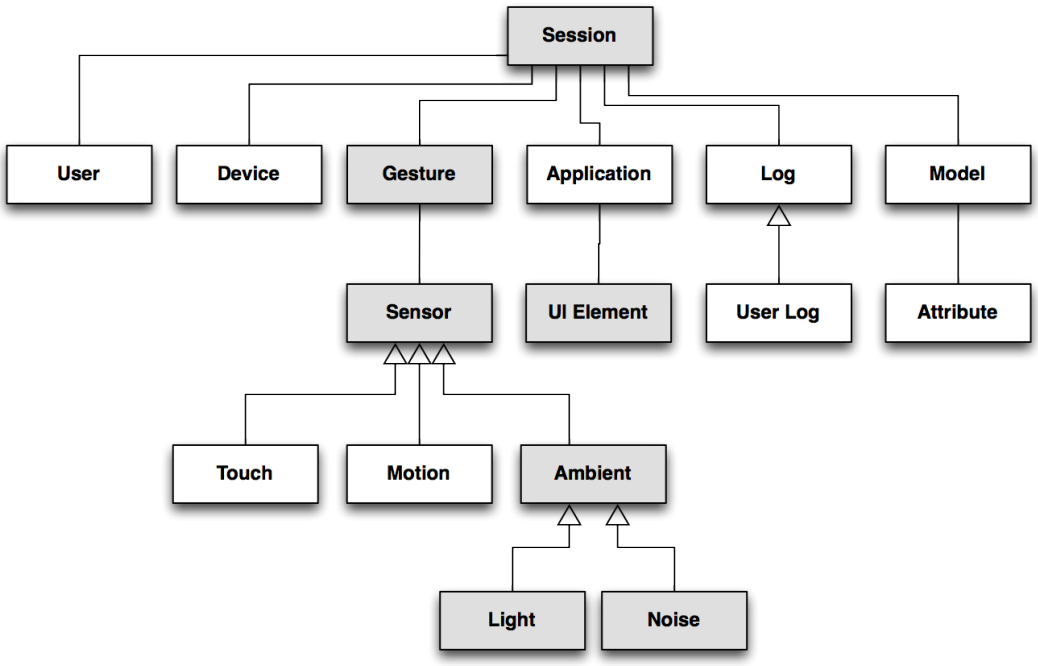

Figure 6.2 UML diagram of the revised SUM database structure used for capturing user interaction in relation to the current session.

Session: to allow SUM to gain an understanding of the variances or similarities of user interaction characteristics across and within usage periods, user interactions are automatically collected and structured into sessions. By defining the usage periods into sessions, SUM enables these interaction periods to be compared and classified to identify the longer-term changes and behaviours. Session measurements will be discussed further within *usage behaviours* (Section 6.2.3).

UI Element: SUMClient now supports the use of GestureRecognisers to collect measurements of user interaction rather than overwriting the interface components themselves. Where previously the SUM framework would have a complete understanding of the interface component being interacted with, using this new approach allows developers to define interface components of which SUM has no prior knowledge. This allows a richer analysis of user interactions in relation to the

specific interface elements, for example: In measuring touch durations within List objects only. SUM needs to capture more details from the application interfaces. When a touch gesture event is 'fired', the framework receives the object relating to the touch properties and the associated interface object. SUM collects the same details as the earlier *Target* object, with the addition of the interface component *type,* and *tag name.*

Sensor: within the current version of the SUMClient, sensor objects all inherit the abstract *sensor* object, enabling gestures to be generically applied to sensor readings from any of the available sensors. The sensor object contains only an *id* and *timestamp.* Other sensors can inherit and extend this object to include the parameters required to capture their own measurements, for example the *Ambient Light and Noise* objects also include an additional field for *level* of light or noise. Sensor objects are intended to represent the lowest level of measurement for a single instance or state of the sensor. To conform with this new structure the *Touch* objects have also been revised in this version.

Touch: previously touches were stored as the complete touch gesture, i.e. a single tap, providing only a single *x,y* location, *target,* and overall *duration.* However, the revised version of this object means that the same single tap is now captured in the following way, as three table entries:

```
Touch{id:1,x:30,y:43,state:begin,target:5,duration:0,timestamp:13704
14769},

Touch{id:2,x:32,y:42,state:move,target:5,duration:.02,timestamp:1370
414769},
```

```
Touch{id:3,x:32,y:42,state:end,target:5,duration:.11,timestamp:13704
14769}
```

The additional *state* field allows the single tap gesture to be split into the individual touch event states, providing a higher level of detail for a deep analysis of the touch characteristics than was previously possible.

Gesture: the gesture object is used to define a series of *sensor* readings that when combined represent the user's interaction, for example the single tap gesture presented above consists of: *touch begin; touch move;* and *touch end* objects. Each one defines the possible states of a touch gesture. Combined they represent the complete sequence of events that define a single tap gesture. Gestures are described as having *start* and *end* times, and a *type.* The type field allows the low level sensor reading to be associated with a high level gesture name, for instance *horizontal swipe.* By defining these types SUM is able to select the appropriate gesture parameters that are of importance to the successful recognition and distinction of the gesture from other similar gestures.

6.2.2.2 Data Synchronisation

Formerly, the synchronisation of user data was performed manually on the device through the built in configurations screens of SUMClient. To support the long-term use and real world application of the SUM framework this process needed to be automated and invisible to the user. Data synchronisation should not disrupt the natural flow and interactions of the user with the device and its applications. However, the conceptual design of SUM requires that application data be pooled within the SUMServer to enable the sharing of interaction data for modelling.

The revised solution to SUM data synchronisation applies a time limit threshold. Figure 6.3 demonstrates the SUM workflow to maintain regular synchronisation of user data. The default threshold was set to 30 minutes, meaning that at the application launch if the last sync time was greater than this threshold then SUMClient would attempt a background sync with SUMServer. An active network connection is required in order for SUM to sync, therefore in periods of no network access SUM will neither sync nor request an updated user model. If the device has an active connection and the SUMServer is reachable, the process will begin. No feedback or alert is presented to the user other than the device's notification of network activity. Not all mobile operating systems support background processing once an application is closed, therefore the SUM synchronisation process could be interrupted if a user were to close the application before it completes. To avoid any loss of valuable user data, SUMClient recognises the application state, and in the event of an application being closed SUMClient pauses the sync and continues again once the application is reopened.

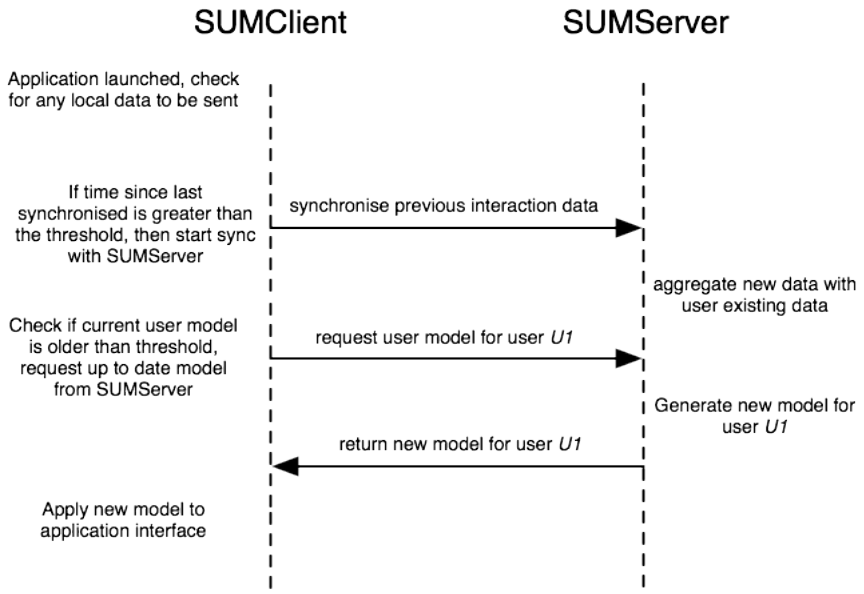

Figure 6.3 Communication workflow between SUMClient and SUMServer for data synchronisation and user model request.

Within the previous laboratory evaluation of SUM (Chapter 5) the researcher encountered syncing problems as a result of the shared server hardware, which had a concomitant effect on the SUMClient synchronisation. The earlier version of SUMClient used synchronous communication calls, which resulted in large periods of time waiting for SUMServer responses thus increasing the overall time to sync up. To address these issues, the revised version makes asynchronous communication calls to the SUMServer alleviating the delays. Furthermore the method for transferring large amounts of device motion data has been refined. Where previously the motion data was sent and stored in real time, accounting for much of the communication bottlenecking, the SUMClient now transfers an entire session's motion data as a single CSV file which the SUMServer processes post-synchronisation.

6.2.3 SUMServer

On the face of things the high level architecture of the SUMServer remains unchanged from the earlier version (Figure 4.4). However, due to the significant changes made to the SUMClient (Section 6.2.2) the internal functionality of the SUMServer has also changed. Where previously the SUMServer would receive the pre-processed gesture data from the SUMClient, this is no longer the case. To allow deeper analysis of the low level behaviours within user interactions, the framework requires SUMClient to collect the sensor measurements in their raw state, and relays the raw data to the SUMServer for processing and analysis. In order to support the increase in the amount of data being captured and transferred by the SUMClient, the web services within SUMServer have been overhauled to streamline and optimise performance as previously mentioned within the SUMClient data synchronisation (Section 6.2.2).

In addition, the SUMServer has also been revised to support remote evaluations spanning longer periods of time in the light of findings from the laboratory user evaluations (Chapter 5). Revisions have been made throughout the SUMServer, both to support the long-term use of the devices and the resources specifically needed/required to assist in the remote evaluation process. This section will now present the processing techniques applied to the interaction data to extract measurement features, followed by a discussion of the provisioned methods for monitoring usage behaviours and long-term remote use.

Formerly the SUM framework modelled the user interactions using the overall *touch duration* and *touch down target offset*. While these individual models provided a

significant improvement on the device's default behaviours (Chapter 5), they neglected to consider the finer levels of the user interactions, for instance, movement, speed and acceleration between touch states. Previous works with mouse cursor input have demonstrated that these features could be used to identify individuals with additional motor ability needs (Hurst, Hudson, Mankoff, & Trewin, 2008a). Hurst et al. (2008a) used three sets of features: task specific features; click specific features; movement related features.

Mouse and touchscreen sensors have nearly identical measurement attributes, and both share similar gestures, for instance the *mouse click* and a *touchscreen tap* gesture.

```
Click{

Mouse{id:1,x:30,y:43,state:down,target:5,duration:0,timestamp:137041
4769},

Mouse{id:2,x:32,y:42,state:move,target:5,duration:.02,timestamp:1370
414769},

Mouse{id:3,x:32,y:42,state:up,target:5,duration:.11,timestamp:137041
4769}

}


Tap{

Touch{id:1,x:30,y:43,state:begin,target:5,duration:0,timestamp:13704
14769},

Touch{id:2,x:32,y:42,state:move,target:5,duration:.02,timestamp:1370
414769},

Touch{id:3,x:32,y:42,state:end,target:5,duration:.11,timestamp:13704
14769}
```

}

Recognising these similarities between the two inputs, SUM proposes the following features to describe touchscreen interactions adopted from the mouse features used by (Hurst, Hudson, Mankoff, & Trewin, 2008a). Figure 6.4 illustrates a touch sensor measurement and details the various states of the touch gesture and the corresponding features that have been defined from them.

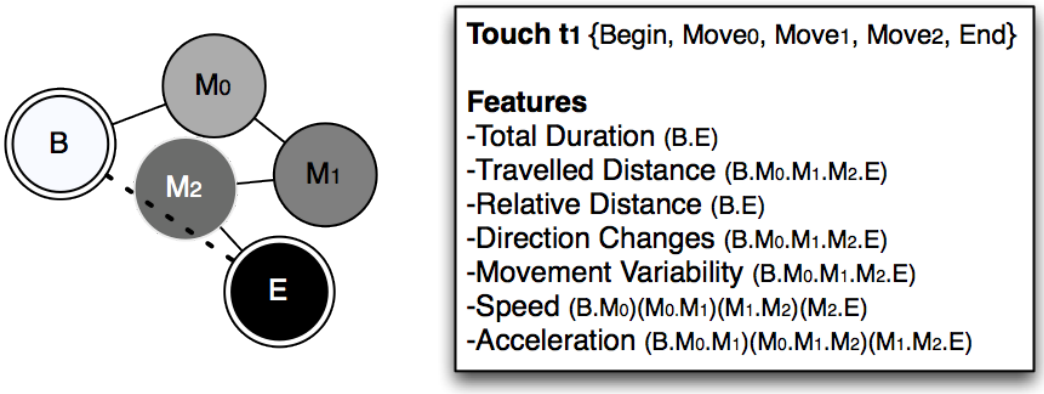

Figure 6.4 SUM touch sensor feature set

The initial user evaluations with SUM (Chapter 5) lasted only one hour at a time, with a period spanning several days or weeks between the two sessions. The researcher observations from that study suggested there could potentially be a high variance in the user's abilities between application usage. That work suggested that measurements with longer collection periods are required to understand the true extent of the variability of an individual's abilities. From participant interviews, it was learned that factors influencing their interactions with devices included underlying symptoms of medical conditions, effects of medications, fatigue and situational or environmental conditions. By grouping user interactions into distinct sessions, SUM is able to extract specific features to describe each session, thereby

allowing the classification and analysis of usage patterns. For example, an individual with Parkinson's disease may experience large amounts of unintentional hand movements in the morning before taking medication, but these symptoms may be greatly reduced by the afternoon. The complete feature set and analysis techniques applied by SUM are presented in Chapter 8.

6.3 Building with SUM

One objective of the SUM framework is to reduce the effort required by developers to create applications that take advantage of the user modelling abilities of SUM. Therefore, much of the functionality of the SUM framework has been automated to enable developers to use SUM without altering their current workflows. Similarly, the SUM framework has been designed such that developers need not be experts of user modelling, accessibility and interface adaptation to benefit from it. To achieve these objectives and ensure that the process of embedding the SUM framework is intuitive and efficient to the development workflow, the method of capturing and responding to user interactions has been updated to leverage the gesture recogniser system now common amongst the mobile operating systems.

6.3.1 Gesture Recognisers

Version one of the SUM framework used overwritten user interface components to capture measurements of user interactions, requiring developers to substitute the OS UIControls for the SUMControls. When using the default UIControls this substitution was straightforward. However, if developers were creating completely customised interface components that didn’t inherit the abstract UIControl class set, then there would be no guarantee that the SUMControls would use the interaction

logic as the developer intended. Version one required the developer to redevelop the controls and include the SUMControl class before the controls would work with the SUM framework.

Since the original development of the SUM framework, many of the mobile touchscreen devices' operating systems have been revised to include *interface listeners* solely responsible for the recognition of user interactions, known as GestureRecognisers. Developers are encouraged to use GestureRecognisers with their interactive interface components rather than implementing the interaction handling code into the interface component itself. GestureRecognisers are supported across interface components, meaning that the same recogniser could be used on a button and an interaction image. Figure 6.5 provides an example of the native iOS code required to create two tap GestureRecognisers, one for a single tap and the other for a two-finger tap. The GestureRecognisers are then assigned to a button interface component. Similarly, Figure 6.6 demonstrates the code required to perform the same task using the SUMClient. There is no alteration to the task workflow since the final syntax of SUM mimics that of the iOS API.

```
///Define a two TapGestureRecognisers.
///SingleTap - successful tap will call the "openURL" method within this class

UITapGestureRecognizer *singleTap = [[UITapGestureRecognizer alloc]initWithTarget:self action:@selector(openURL)];
[singleTap setNumberOfTapsRequired:1];


///TwoFingerTap - successful two finger tap will call the "editURL" methid within this class

UITapGestureRecognizer *twofinger =  [[UITapGestureRecognizer alloc]initWithTarget:self action:@selector(editURL)];
[singleTap setNumberOfTapsRequired:1];
[twofinger setNumberOfTouchesRequired:2];


///Define a custom UIButton

UIButton *button = [UIButton buttonWithType:UIButtonTypeCustom];
[button setTitle:@"Interact with me" forState:UIControlStateNormal];


///Set the gesture recognisers of the button to include twofinger and singleTap

[button setGestureRecognizers:[NSArray arrayWithObjects:twofinger,singleTap,nil]];
```

Figure 6.5 Sample code for creating two iOS TapGestureRecognisers for a single tap, and two finger tap, then creating a custom button that responds to those gestures.

```
///Define a two SUMTapGestureRecognisers.
///SingleTap - successful tap will call the "openURL" method within this class

SUMTapGestureRecognizer *singleTap =  [SUMGestures tapWithTarget:self action:@selector(openURL)];
[singleTap setNumberOfTapsRequired:1];


///TwoFingerTap - successful two finger tap will call the "editURL" methid within this class

SUMTapGestureRecognizer *twofinger =  [SUMGestures tapWithTarget:self action:@selector(editURL)];
[twofinger setNumberOfTapsRequired:1];
[twofinger setNumberOfTouchesRequired:2];


///Define a custom UIButton

UIButton *button = [UIButton buttonWithType:UIButtonTypeCustom];
[button setTitle:@"Interact with me" forState:UIControlStateNormal];


///Set the gesture recognisers of the button to include twofinger and singleTap

[button setGestureRecognizers:[NSArray arrayWithObjects:twofinger,singleTap,nil]];
```

Figure 6.6 Sample code for creating two SUMTapGestureRecognisers for a single tap, and two finger tap, then creating a custom button that responds to those gestures.

The SUMGestureRecogniser provides all of the functionality and control available within the OS default gesture recogniser, behaving no differently than the stock API versions. Typically when a developer defines a GestureRecogniser and releases the application, the recogniser behaviours remain static and software updates are required to alter their behaviours.

However, using SUM, once the development is complete and users are interacting with the applications, SUMGestureRecognisers begin to mould themselves to the

abilities and behaviours of the user. Figure 6.7 details the process applied by the SUMClient to provide interface adaptations to GestureRecognisers in order to create ability-based interactions. SUM allows the software developer to design gesture interactions for an "average user", and then provide individuals with a personalised interaction matching their needs, abilities and interaction style.

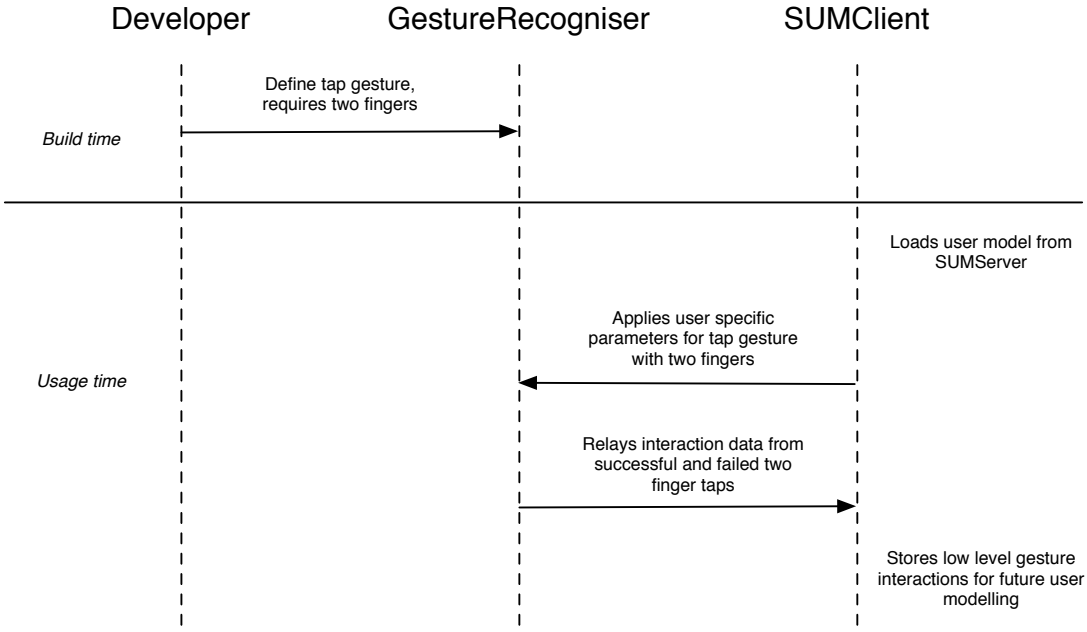

Figure 6.7 Adaptation of gesture recognisers by the SUMClient at usage time within the application

6.4 Conclusions

This chapter has detailed the revised design of the SUM framework to address limitations of version 1 (Chapter 4), and extend the functionality to achieve more granular measurements of user interactions with mobile touchscreen devices than version 1, and support longer user evaluation periods. The chapter discussed the motivation and rationale for the redesign of the framework, detailing the benefits for the accuracy of the interaction data collection. The chapter detailed the methods for collecting usage behaviour and supporting long-term user evaluations, as well as the

statistical modelling approach used by SUM. The chapter then presented the updated procedure for embedding the SUMClient within applications, highlighting the simplicity of the revised API versus the more complex earlier version.

Chapter 7. In-Situ User Study with SUM

This chapter describes an in-situ user study of three mobile applications in which the revised SUM framework is embedded (Chapter 6). The study involved 12 participants with visual and motor impairments, and spanned a four-week period. The primary objective of this evaluation was to capture measurements of user touchscreen performance within a real-world context to expose the usage behaviours and fluctuations of user performance and ability. Secondly, this evaluation aimed to explore the potential of the SUM framework as a method of capturing and measuring user touchscreen performance from the wild. Therefore, no interface adaptation were applied during this user study, instead the SUM framework was used to simply collect the user interactions throughout the four-week period. The chapter begins by discussing the challenges and benefits of conducting user studies outside of the controlled laboratory environment. Next the chapter will present the user study design consisting of: a description of the participants taking part in the study; an overview of the study apparatus and experimental applications, together with an outline of the procedure applied, including the measurements resulting from the evaluation. These results are reviewed in relation to the collective and individual usage and interaction behaviours of the participants, and are followed up with a discussion of the individual differences in abilities and needs of the twelve participants.

7.1 From the Laboratory to the Real World

The laboratory based evaluation of the SUM Framework (Chapter 5) represented a proof of concept, and served as a pivotal assessment of both the modelling and

adaptation techniques of SUM to support the needs of a diverse group of users. As previously stated in Chapter 5, the laboratory based user studies were divided into two sessions of one hour, users tested the three applications across both sessions. The user evaluations included external assessments conducted by the researcher to acquire additional information for the user models, and models were also refined based on the observations made within the evaluations. Expanding on the knowledge gained from this initial laboratory study (Chapter 5), revisions were outlined to further develop the SUM Framework to reduce the reliance on observed measurements, and address the limitations of the first version of the framework. These revisions were covered in detail previously within this thesis (Chapter 6).

The objectives of this in-situ study were primarily to address the three key limitations of the laboratory study:

Snapshot measurements: Participants reported on a snapshot effect when measuring their abilities through the short laboratory studies. Many participants discussed the noticeable differences in their abilities resulting from medication cycles, which were not captured during these evaluation sessions. Short evaluation sessions of approximately one hour separated by longer time intervals will produce *snapshots* of a user's interaction behaviours and abilities, and create user models that are susceptible to extreme skewing.

Unnatural Behaviours: Laboratory based evaluations do not reflect users' real-world interaction behaviours. The user is placed in an unfamiliar and potentially intimidating environment, then requested to perform very specific tasks under the premise they are being observed and monitored. While the stimulus and interaction

constraints within the laboratory study were designed to reflect typical mobile applications and their use, the user behaviours did not necessarily represent the user's real-world actions. For example, *P7* used a magnifying glass at home to view small printed text on his mobile phone; this was not possible within the laboratory environment.

Loss of Detail: The raw interaction data was processed within the SUMClient to produce recordings of individual gestures e.g. single tap, vertical swipe. This allowed the SUMClient to retain an overview of each individual interaction, while reducing the required storage capacity and network usage by not retaining each internal state of the gesture e.g. *touch begin, touch move, touch end.* However, this resulted in the loss of detail and granularity of touchscreen interaction characteristics. The SUMServer was unable to explore the individual state that made up the gestures, thus limiting the potential analysis and modelling features post-study.

While the laboratory user evaluations provided accurate measurements of individuals' interaction abilities at that instance in time, it was important to define a study design that could obtain continuous measurements of the users across longer periods of time to understand how their interaction abilities fluctuated. Similarly, to address the unnatural behaviours within the laboratory setting, the study design needed to remove the necessity for laboratory sessions and allow users to freely interact with the technologies in real-world settings. Finally to guarantee that this evaluation captured user interactions at a high enough level of detail, revisions were made to the SUM framework (Chapter 6), ensuring that the raw touchscreen

interactions are captured with finer granularity than previously used in the laboratory study (Chapter 5).

Given the new requirements for longer collection periods outside the controlled laboratory, two possible approaches were considered to further evaluate the SUM framework's ability to model user touchscreen performance and needs.

- A controlled study with a longer collection period to capture repeated measures from a concentrated population.
- Large-scale marketplace study with a much wider population.

The latter option could have been achieved by leveraging the popularity of mobile app-stores to reach larger participation numbers, similar to the approach of (Henze et al., 2011). However, while the (Henze et al., 2011) study was able to attract 91,731 installations, contributing to over 120 million touch events (touch begin, move and end is one event in this case) the authors had no control over the characteristics of the participants. In fact, regardless of complex individual characteristics, the authors had no method of tracking a single user within the study; instead the independent measures were based on the installations. However, since a user could have had multiple installations, or multiple users could have played a single installation, the measurements do not reflect the individual measurements of a single user. Furthermore, Henze et al. (2011) reported an average of 1315 touches per installation; applying their logic of *"one touch event per second"* this equates to an average collection period of 22 minutes per installation. Using this approach within the SUM study would have resulted in similar limitations to the laboratory SUM

evaluation, with lack of interaction evidence per user over a longer period of time. The approach of (Henze et al., 2011) produces a more focused high volume understanding of the device behaviours and less of an understanding of individual user abilities.

For these reasons the app-store approach was rejected in favour of the controlled study spanning four weeks, with a much smaller sample size of just 12 participants. This design allowed the researcher to control the participant recruitment process thus helping to remove one of the many uncertainties when conducting in-situ evaluations. Pre- and post-study informal discussions with each participant were also used to aid the analysis of the evaluation data, which would not have been feasible using the app-store approach.

7.2 User Study

This section presents the design of the in-situ user evaluation using the revised SUM framework (Chapter 6), involving 12 individuals with diverse levels of visual and motor abilities.

7.2.1 Participants

Twelve participants, seven females and five males, took part in the user study. They ranged in age from 21-75 (M=55, SD=20) years old. All participants exhibited abilities that qualified them as having a motor and/or a visual impairment. In addition, all participants were required to own and use a mobile phone (although not necessarily a smartphone) and to have a home WiFi connection to the Internet in order to ensure that the study devices could regularly communicate with the remote server. Table 7.1 provides information about the participating individuals and their characteristics.

Once invited into the study, participants were informed that they would be allowed to keep the mobile device upon completing the study. The researchers hoped this would encourage the participants to explore the devices, and integrate their functionality within their daily lives. See appendix 3 for information sheet and consent forms.

ID	Age	Gender	Touchscreen Experience	Group	Impairment	Current Accommodations
P1	55	Female	Self-service machines	Motor Control	Parkinson's Disease, slight hand tremors	Regular medication to suppress symptoms
P2	59	Male	Tried iPod touch before.	Motor Control	Spinal injury, muscle spasms, hand tremors, Sensitive to light	Regular medication to suppress symptoms
P3	57	Male	Self-service machines	Motor Control	Parkinson's disease, hand tremors	Regular medication to suppress symptoms
P4	67	Female	Tried iPod touch before.	Blind	Retinal detachment, macular degeneration, diplopia	Guide dog, magnifying glasses, screen reader software on PC
P5	73	Female	Tried iPod touch before.	Motor Control	Myalgic Encephalomyelitis. Muscle twitches and spasms in arms and hands..	Medication to suppress mobility symptoms, not cognitive
P6	22	Female	Has an iPod Touch	Motor Control	Hypermobility syndrome, locking joints and tremors in hands	Wheelchair, medication to suppress symptoms
P7	63	Male	None	Motor Control	Parkinson's disease, hand tremors	Regular medication to suppress symptoms
P8	21	Female	Tried iPod touch before.	Motor Control	Essential tremor	Medication when symptoms increase
P9	65	Female	Tried iPod touch before.	Motor Control	Parkinson's disease	Originally medications. During the study underwent Deep Brain Stimulation (DBS) surgery
P10	24	Male	Tried iPod touch before.	Blind	Registered blind, issues adjusting to changes in light levels	Monocular, screen magnification on PC, mobile with large buttons.
P11	75	Male	Has an iPod Touch	Motor Control	Parkinson's disease, hand tremors	Medication when symptoms increase
P12	74	Female	Tried iPod touch before.	Motor Control	Essential tremor	Regular medication to suppress symptoms

Table 7.1 Participant profile; dominant hand used when interacting with the device; stereotypical disability grouping associated with participant; specific impairment and current accommodations to deal with symptoms.

7.2.2 Apparatus

The purpose of this evaluation was to capture the real-world interactions of the participant when using the mobile touchscreen devices, in order to better understand how an individual's abilities and interaction characteristics vary, thus allowing us to refine the processes needed to model these changes. The high-level structure of the apparatus mirrors that of the laboratory evaluation (Chapter 5), whereby the user was provided with a touchscreen device preloaded with the stimulus applications. These experimental applications were designed to capture the user's interactions, and relay the data back to the centralised SUMServer through the participant's home WiFi network. The SUMClient was embedded into each application, and is responsible for handling the data synchronisation process, as discussed previously (Chapter 4).

Participants were each provided with iPod touch devices, as used within the laboratory study. However, in the laboratory study the second Generation devices were used, in this study these were exchanged for the fourth Generation device, running the iOS 6.1, rather than iOS 3.0 as used in the previous study. The fourth generation devices have an almost identical external look and feel as the second generation device, although they possess additional sensors and hardware upgrades. Table 7.2 provides an overview of the significant changes between the second and fourth generation iPod touch devices.

Generation	2nd	4th
RAM	128MB	256MB
CPU	533MHz	800MHz
Camera	N/A	Front & Back
Battery	739 mA-h	930 mA-h
Weight	115 g	101 g
Microphone	Yes	Yes
Accelerometer	3-Axis	3-Axis
Gyroscope	N/A	3-Axis
Vibration Motor	N/A	N/A
Screen Resolution	320x480	640x960
Pixels per inch	163	326
Screen Dimensions	74 mm (H) 49 mm (W)	74 mm (H) 49 mm (W)
Device Dimensions	110 mm (H) 61.8 mm (W) 8.5 mm (D)	110 mm (H) 58 mm (W) 7.1 mm (D)
Operating System	iOS 3.0	iOS 6.1

Table 7.2 iPod touch comparison table of second and fourth generation devices (Wikipedia, n.d.).

While the screen resolution has been doubled on the fourth generation device, the physical dimensions of the screen remain unchanged and this increase in pixels results in a sharper screen definition. Likewise, from the programming perspective developers define the interface elements' dimensions with the original pixel sizes, and the dimensions will be automatically doubled at runtime to accommodate the new screen resolutions. For example, a button with the screen location and bounding dimensions of (10,10,300,60) would be mapped to (20,20,600,120) within the new screen resolution. Throughout this thesis the sizes are discussed using the original pixel resolutions for consistency with the previous chapters.

The server hardware configuration remained unchanged from the laboratory evaluation. However, the server was updated with the revised version of the

SUMServer framework (Chapter 6). In addition to the changes described within the revised framework chapter, further tools were developed to support the researchers with the running of the in-situ user study:

Over-the-air-updates: All of the experimental apps were embedded with TestFlight[3] , a commercial beta testing system for mobile application development. A key feature of the TestFlight framework is the ability to update application versions remotely, allowing the researchers to provide over-the-air-updates to participants should issues arise regarding the applications being used. One of the challenges of putting new devices out in the field is the unpredictable nature of the interactions resulting from users exploring a new device. The design of the iOS system is such that third party applications cannot be *locked* or *secured* to restrict the removal of the application. As a result of this lack of functionality it meant that participants were free to remove the three experimental applications during the study. Participants *P11* and *P12* were exploring the devices and accidentally removed the Sudoku application, along with any unsynchronised interaction data from the application. The TestFlight system was able to support the reinstallation of the Sudoku application on both devices, allowing the participants to continue with the study that afternoon.

Status and Usage Tracker: The researchers were able to log in to secure web pages and retrieve feedback on the status of the SUMServer to verify its functionality and network connectivity. This enabled the researchers to promptly identify and respond

3 https://testflightapp.com/

to technical issues within the SUMServer from anywhere at any time. Similarly secure web pages were defined to display each participant's usage information, giving the researcher an overview of the last time a participant accessed an application and for how long from the SUM synchronisation data, notifying the researcher to participants' activity or inactivity with the applications. The researcher could then contact the participant to offer support, answer queries and resolve any potential technical issues with the devices, or encourage him/her to engage with the applications more often. The devices required an active WiFi network to access the internet and sync with the SUMServer. Some participants were using the applications but were unaware that they were not connected to their home WiFi. The researcher interpreted this as inactivity and was able to resolve the problem with the user.

7.2.2.1 Experimental Apps

The purpose of this study was to capture touchscreen interactions of the participants when naturally using touchscreen devices in the real-world; therefore the applications performed no interface adaptations or personalisation. Each application used the SUMGestureRecognisers to capture and interpret the user's touchscreen interactions. However, no touch models were applied to the applications thus the SUMGestureRecognisers behaved as the default iOS UIGestureRecognisers would have e.g. UITapGestureRecogniser for single taps and UISwipeGestureRecogniser for scroll and swipe gestures. The gesture classifications provided by these gesture recognisers were used to define the classification of the touch interaction data. In the event that a user's touch interactions were not recognised, the touch data was captured and classified as an 'Unrecognised' gesture.

Due to the limitations of the developer API access and sandboxed design of the iOS platform, it was not possible to simply embed a background service to capture all user touch interactions from any installed application. In order to collect the users' touch interactions in relation to the applications and interface components it was necessary to develop experimental applications to be installed on the device.

As part of a previous laboratory evaluation (Chapter 5), participants were asked to think of any daily tasks or activities they might like to carry out using a touchscreen device. The top suggestions included medication reminders, notes and lists for shopping, checking emails, playing games, TV listings, and browsing the web. From these suggestions three applications were identified – Memo, Sudoku, and TV Guide. This selection was also based on considerations of coverage of potential interface gestures and probability, frequency, and temporal distribution of use.

The feature sets and interface designs for the three applications were based on similar applications within the App Store[4] to ensure their relevance and to reduce potential design bias by the researchers. The applications make use of a number of the traditional touchscreen interface components including:

- Table views
- Date Pickers
- Switches
- Buttons

4 http://itunes.apple.com/gb/genre/ios/

- Number pad
- Text views
- Navigation bars

7.2.2.2 Memo

The Memo application represents a combination of the requests by participants for a method to receive medication reminders and create notes or lists for shopping. Participants added new reminders using the Add memo interface as illustrated in Figure 7.1. Participants could give an item a title and additional details using the standard iOS onscreen keyboard, with the option of setting a due date and reminder alert. Once added, the new items appear within the four tabbed views *Today*, *Week*, *All*, and *Complete*. The memos can be edited or marked as complete by tapping the list item, presenting the user with an interface similar to the Add Memo screen. Alternatively participants could quickly mark a memo as checked or unchecked by performing a horizontal swipe from left to right on the item.

The Memo application enabled participants to set reminders for items, and receive device notifications at the scheduled date and times, which was useful for reminders relating to medication times, or appointments. It was expected that participants' interactions with the Memo application would be relatively short, either adding a new item or simply responding to a reminder notification. Likewise, it was believed that these interactions could take place at various times throughout the day and night, spanning the four-week long period.

Figure 7.1 Memo application for iOS devices. Add memo screen (A) and Memo list showing this week (B).

7.2.2.3 Sudoku

Many of the participants mentioned their enjoyment of crosswords and Sudoku puzzles in their newspaper. Either of these would have made a suitable game application for this study, as they both require participants to select squares from a board grid and enter values. Sudoku was selected, as relatively straightforward algorithms could be used to generate new puzzles on request, and the board design could fill the entire space of the screen. In comparison, crossword puzzles would have had to be manually created in advance and additional space would need to have been provided for the crossword clues (something better suited to a larger tablet device). The basic functionality of the Sudoku application allowed participants to select a New Game, and then select a game difficulty level of easy, medium or hard. This would then generate a new Sudoku puzzle at the selected level (based on the number of empty squares the puzzle starts with). Figure 7.2 shows the Sudoku board

presented to a participant upon starting a new game. The board allowed participants to select a target square by tapping it, causing the number pad to appear as illustrated in Figure 7.2. Values could be entered through tapping on the required number from the number pad. The number pad also contained a *Clear* button to remove a previously entered value, and a *Hide* button to remove the onscreen number pad and reveal the entire board again. The Sudoku board would also respond to a two-finger pinch gesture, allowing the participant to zoom in and out on the Sudoku board.

The Sudoku application included provided participants with a description of Sudoku and detailed how to play, which could all be accessed through the 'about' section of the application. Finally, the application included a *Task List* option, which contained a list of 14 predefined Sudoku puzzles of varying difficulty. The reasons for including the Task List of predefined puzzles are discussed in Section 7.2.3

It was predicted that participants would have longer interaction sessions with the Sudoku application than with the Memo and TV Guide applications. Sudoku was also predicted to provide the highest volume of touch interactions, with each game consisting of between 40 and 70 interactions, depending on difficulty level. The application encouraged participants to make bursts of accurate target selections over an extended period. Again, it was hoped that participants would enjoy playing a short Sudoku puzzle at various times during the day, helping to provide good coverage of the participants' varying abilities over time.

Figure 7.2 Sudoku application for iOS devices. Gameplay screens showing the Sudoku board (A) and board with open keyboard (B).

7.2.2.4 TV Guide (version 2.0)

The earlier laboratory study (Chapter 5) included a TV Guide application with a preloaded set of programme listings, used during the tasks designed to simulate TV listing search and browsing. Based on the observations and participant feedback from this study, the TV Guide application seemed to be a welcome alternative to the traditional paper-based and TV-based electronic programming guides. The application made use of the software platform's table views to present the TV and radio programmes using a list-based navigation method.

Extending the functionality from the first version of the application, the revised TV Guide app downloaded daily TV and Radio listings from online sources[5] and stored

5 http://bleb.org/tv/data/listings/

them locally. Participants could navigate the content by TV Channel, Radio Station, or by Programme title; a sub-navigation then allowed participants to view lists for today, tomorrow or A-Z (over both days). An example of the Today listings' screen is shown in Figure 7.3. The application would only display programmes that were currently airing, or scheduled to air soon (within a 48 hour period). Typically the current programme would appear at the top of the list, with the exception of shows being viewed using the A-Z option.

Participants using the TV Guide application could navigate by performing a vertical swipe gesture to scroll through the listings, then use a single tap gesture on the desired item to select it; an interaction style found in similar table view interfaces. Once a programme had been selected, the application would display the full description for that item (Figure 7.3), providing the user with the programme title; episode information; description; channel number; access formats (if the show contains subtitles, audio description or sign language); and finally, an option to set a reminder alert for that programme. By tapping the "Set Reminder" button, participants could schedule a notification alert for five minutes before the programme was scheduled to air.

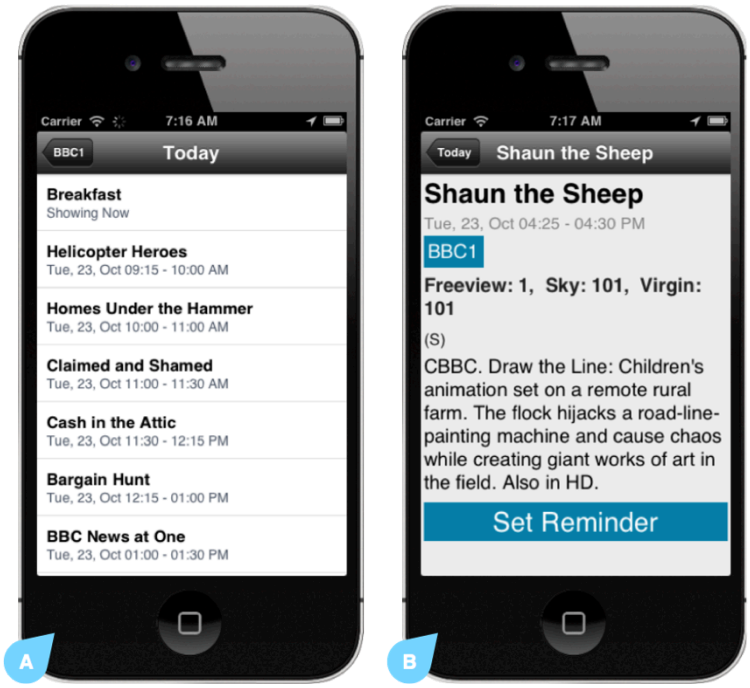

Figure 7.3 TV Guide application version 2 for iOS devices. List of today's programmes on BBC1 (A) and details view for "Shaun the Sheep" TV programme (B).

The interface style of the TV Guide application was similar to that of other information retrieval applications such as email, blog or news feed readers and used user interface controls similar to those used in the built-in iOS applications such as Contacts and Settings. The researchers expected that participants would only access this application at particular points within the day, prior to and during TV viewing periods.

7.2.3 Procedure

The evaluation consisted of three stages: an initial training session and informal discussion with the researcher; the four week application use in the wild; and a post-study discussion with the researcher. Participants were provided with additional application example sheets (Appendices 4, 5, 6), demonstrating the typical interaction scenarios of each application. These included sample memo entries, TV

guide queries and Sudoku solutions to support the participants throughout the evaluation. Upon completion of the study participants were gifted the touchscreen devices as a thank you for their participation within the evaluation study. This section discusses the three stages of the evaluation:

7.2.3.1 Training Session

The principal researcher met with each of the participants at the beginning of the four-week evaluation for a 30 to 40 minute session to introduce the purpose of the research and demonstrate each of the three applications. Most participants were able to visit the University of Dundee to complete this training session, however, the researcher did visit three of the participants at home. Participants P1, P3, P9 and P12 had never used smartphone devices before, and were provided further training on the basic device functionality and controls within this session. Once the participant felt confident enough to operate the device and the three applications, the researcher entered the unique login details for that participant and activated the SUMClient logging capabilities.

Participants were provided with information to assist in connecting the devices to their home WiFi network in order to ensure that the TV Guide application could download new content and captured interaction content could be synchronised with the SUMServer. Printed copies of the application example sheets were given to each participant. While the participants were aware of the three applications prior to agreeing to take part in the study, it was understood that they might not find the opportunity to engage with the applications every day. The example sheets were designed to encourage and support the participants when using the applications,

providing suggested entries for the memo application, or possible TV channels and programmes to look up. The Sudoku example sheet was slightly different, in recognition of the fact that not everyone is familiar with the game, or is skilled enough to play Sudoku comfortably. Therefore the participants were provided with 14 complete solutions to the 14 preloaded puzzles within the 'Task List' section of the application. These were designed to support the participants through a game of Sudoku until they were confident enough to play a game without help.

Finally, participants were informed that they would be gifted the mobile device upon completion of the study. All participants were encouraged to explore the other applications and device functions, but reminded that the only applications that would capture data were the study applications placed along the bottom row of the app launcher screen.

7.2.3.2 Into the Wild

The three applications would automatically synchronise new session data in the background when applications were reopened and connected to the Internet. The synchronisation process involved transmitting any unsent touch gestures, motion data, application logs and session attributes as raw data through the SUMServer web services. Local data was then marked as sent but not deleted, serving as a backup for the aggregated data collection within the server, in case of network connection errors or other failures. Due to the limitations of the iOS operating system, application data could only be transferred during periods of application activity. Therefore communication would be broken when participants exited the applications during the SUM synchronisation.

Because participants were not in regular contact with the researcher throughout the evaluation, it was crucial that the applications were able to notify the researcher of any communication problems or application errors; this was achieved using the *status and usage tracker* tool which allowed the researcher to view a complete study synchronisation overview for each participant and each application, providing the researcher with the participant id, application, and last accessed time. This allowed the researcher to quickly identify potential application issues, or to contact participants if long periods of inactivity occurred.

Finally, participants were asked to keep a brief diary of their experience during the time of the study. This was aimed at supporting the interpretation of the interactions, in particular providing a better understanding of extreme outliers. Since the system automatically recorded timestamps for all interactions, the participants did not need to keep daily logs of all device use. Instead they were encouraged to take note of the unusual or out of the ordinary situations and behaviours such as feeling very poorly, experiencing extreme symptoms, or travelling somewhere with the device. The applications were embedded with facilities to capture user feedback; however participants opted not to use these, and instead provided paper or emailed diaries following the study.

7.2.3.3 Post-Study Feedback

At the beginning of the study, participants scheduled dates roughly four weeks later to meet with the researchers for 30-minute informal discussion and debriefing. Due to unforeseen circumstances, five of the meetings had to be rescheduled, resulting in longer collection periods for some participants. The dicussions were recorded and

later transcribed by the researchers, allowing the participants to speak naturally about their experiences of the study without interruption or pauses.

7.3 Results

The primary objective of this study was to capture measurements of real-world touchscreen interactions by individuals with visual and motor impairments from the wild. This study explored the use of three mobile applications embedded with the SUM framework: to obtain detailed measurements of touchscreen interactions, identify and understand natural device usage behaviours and interactions. The results presented were fourfold: qualitative behaviours and usage, quantitative interaction measurements, general interaction behaviours and individual interaction measurements.

7.3.1 Qualitative

Qualitative measures were captured through informal dicussions, and provide support and context for the performance measures and behaviours recorded through the application interactions.

7.3.1.1 Applications vs. Tasks

The fundamental goal of this research has been to develop techniques whereby performance measurements could be collected with little or no disruption to users' natural daily lives. Our experimental applications grew from an analysis of what constituted appealing uses of mobile devices for our target population. As a result, the applications were not viewed as tasks or exercises but rather useful tools and fun diversions. During the post-study discussion, participants were asked to "describe your experience of taking part within this user evaluation". All but P10 responded by

commenting on how fun it was to play the game, how handy the TV reminders were, or how useful the memo application was. P10 explained “none of them [applications] were really things that I would do in my normal day. So in terms of that I had to make an effort to use them”. There were however variations in the application usage patterns of the 12 participants, for example, P1 avoided the memo application, so the applications on offer were not desired by everyone. On the other hand, upon completing the evaluations, participants P4 and P5 requested copies of the Sudoku application to continue playing beyond the study, expressing their rekindled enjoyment for Sudoku puzzles since participating in the evaluation. All but P10 remained extremely positive regarding the study activities, and did not associate their device interactions as being measures of performance or evaluation exercises, rather viewed the experience as just “playing with the device” and saw the study as an opportunity to play with a new technology. P8 commented that “the implications for the real world when people are actually using them will be really helpful, because people won’t be using [applications] because they have to, but because they want to [use the applications]”.

7.3.1.2 External Constraints and Factors

The design of the study was for participants to complete a four-week, in-situ evaluation with the mobile device. However, as a result of unforeseen medical conditions, three participants were taken into hospital, and were unable to meet with researchers as originally scheduled and therefore opted to extend their participation. Fortunately all of the participants were fit and able to continue the evaluation once released from hospital care. P6 took the device into hospital for the duration of her stay, and was able to use the Sudoku and Memo applications (since they did not

require WiFi connections). As a result of the design of the SUM framework, all of the interactions made during this extended offline period of use were captured locally and synchronised once connected to an active internet connection.

Two weeks into the study participants P11 and P12 encountered issues with the general operation of the iPod touch devices. P12 accidentally removed the experimental applications from both his device and his wife's (P11) device, when attempting to synchronise their own music collection with the device. Although the SUM framework maintains a local database of interaction data, the stored location of this database is constrained by the platform. iOS forces a highly sandboxed structure on third party applications with all application resources, including databases, being stored within the application package. Therefore, when the applications were removed from P11 and P12's devices so was all of the local data that was yet to be transferred back to the server, resulting in substantial data loss. While this storage restriction exists for iOS, other platforms such as the Android platform allow databases to exist externally to the application package, even within external SD card directories, a design much more fit for this type of purpose.

7.3.1.3 Holding Configurations

Although laboratory based studies have investigated the use of device motion sensors to measure tremor peak movement frequencies and magnitudes from participants with motor impairments (Nicolau & Jorge, 2012b), participant feedback from this in-situ evaluation suggests that this approach may not be reliable within real world contexts. Participants were asked to reflect on their holding configurations within the post-study discussion; P8's response reflected the feelings of some of the

other participants with the comment that: “I’d probably, I’d usually, put it down on the table or on my lap, because obviously you've got the added: ‘if this hand has got a tremor and this hand has got a tremor, and you’re holding it’ [gestures with her hands moving in opposing motions] But if you put it down on a hard surface you've got more [stability].”

7.3.2 Quantitative

All of the interaction data was collected using the SUM framework. This provided a low-level user interaction log, but no high-level domain knowledge or understanding of the applications. However, using the SUM framework the researcher was provided with application logs containing timestamps, and navigation actions such as page loads along with the page titles. Similarly these logs captured application specific interactions e.g. “Cell cleared” from the Sudoku application. The application logs were used to support the analysis of the low-level interaction data.

7.3.2.1 Interaction Behaviours

One of the biggest challenges of analysing performance measurements from individuals with visual and motor impairments is the highly variable nature of their abilities. (Hurst, Mankoff, & Hudson, 2008b) reported large variances both between and within subjects across all performance measures of mouse pointing performance over multiple login sessions. Kruskal-Wallis tests were run to determine if there are differences in touch interaction characteristics between participants. Touch interaction characteristics included the touch x and y offsets, duration and touch movements of tap gestures. Touch interaction characteristics were statistically significantly different between participants, x offset $\chi^2(11)=1483.59$, $p<.001$,

illustrated in Figure 7.4; y offset $\chi^2(11)=1995.81$, $p<.001$, illustrated in Figure 7.5; duration, $\chi^2(11)=2142.09$, $p<.001$, illustrated in Figure 7.6; movement, $\chi^2(11)=110.948$, $p<.001$, illustrated in Figure 7.7.

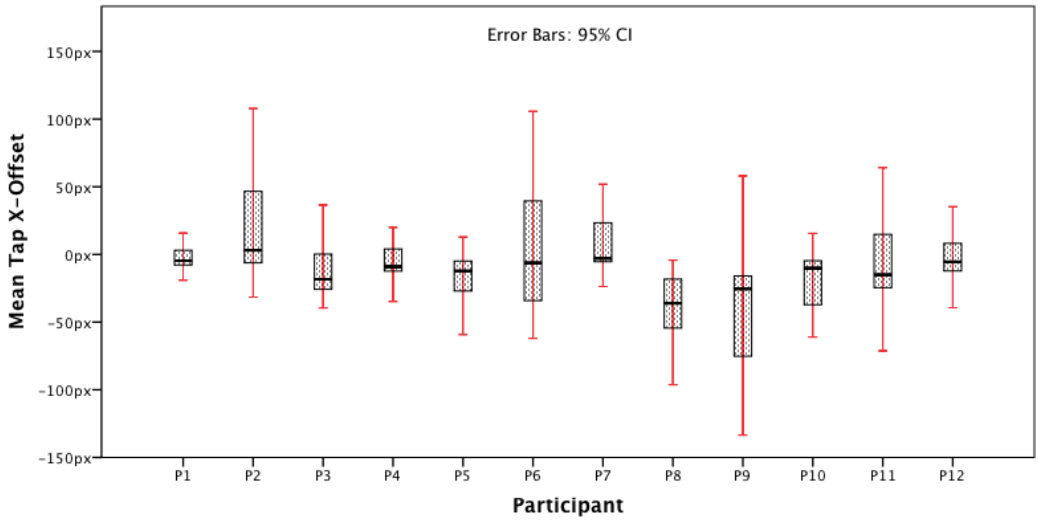

Figure 7.4 Boxplot showing the overall mean x-offsets of tap gestures, per participant. Where values < 0px are offsets left of the target centre, and values > 0px are right of the target centre.

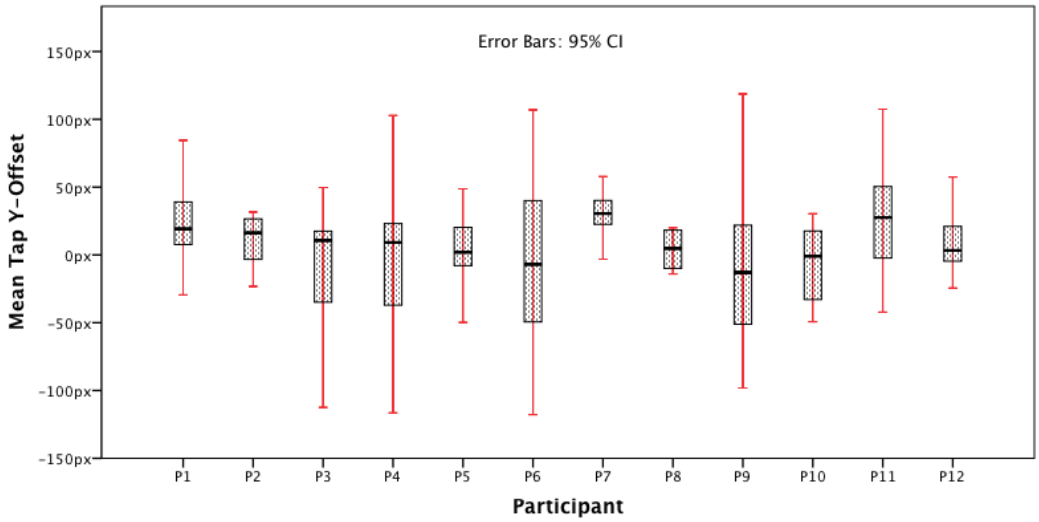

Figure 7.5 Boxplot showing the overall mean y-offsets of tap gestures, per participant. Where values < 0px are offsets above the target centre, and values > 0px are below the target centre.

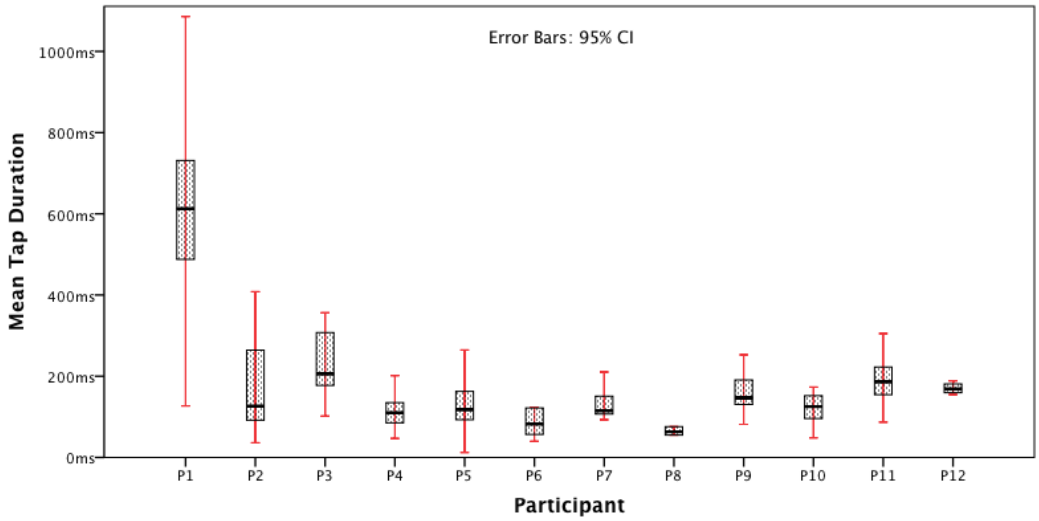

Figure 7.6 Boxplot showing the overall mean tap duration of tap gestures, per participant throughout the in-situ study.

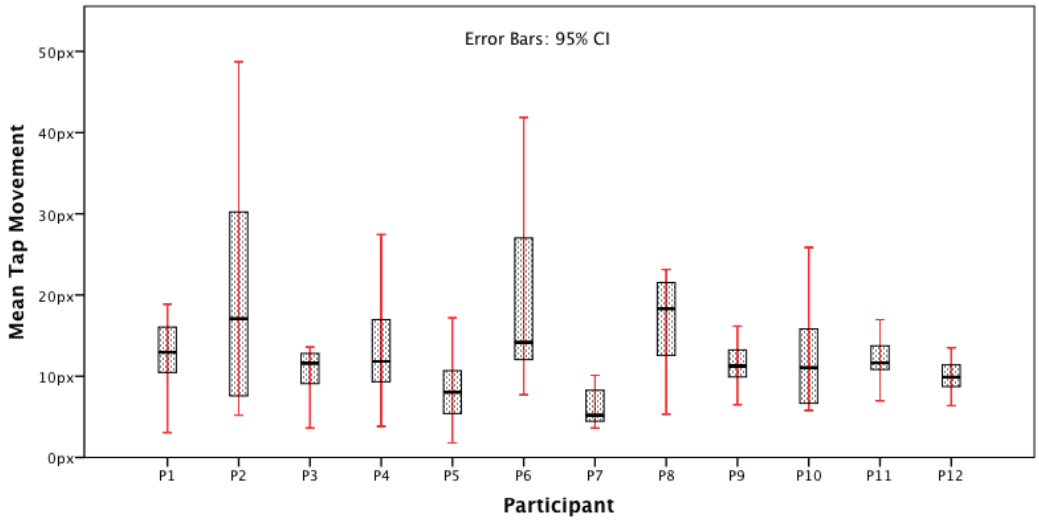

Figure 7.7 Boxplot showing the overall mean tap movement of tap gestures, per participant throughout the in-situ study.

Kruskal-Wallis tests were run to determine if there are differences in touch interaction characteristics between a participant's interaction sessions. All touch interaction characteristics were statistically significantly different between

interaction sessions for participants *P2, P3, P4, P5* and *P9,* (p<.001). No significant differences were observed in touch movement of tap gestures for participants *P1, P7* and *P11* between sessions*P1, P7* and *P11* between sessions, (p>.05). However, the remaining touch interaction characteristics (touch x and y offsets) were statistically significantly different between sessions, (p<.001). Statistical differences in tap duration and movement only were observed between sessions for participants *P8, P10 and P12* (p<.001). Finally, no statistically significant differences were observed in touch interaction characteristics between sessions for participant *P6* (p>.05). Figure 7.8, Figure 7.9, Figure 7.10 and Figure 7.11illustrate the individual participant's daily average x-offset, y-offset, duration and movement behaviours when performing tap gestures. It is clear from these figures that for most participants these interaction characteristics vary dramatically between daily interactions, making it unrealistic to predict from previous sessions. Furthermore, these figures show that while participant's interaction characteristics were significantly different overall, there are sessions whereby two or more participants share similar interaction behaviours and abilities.

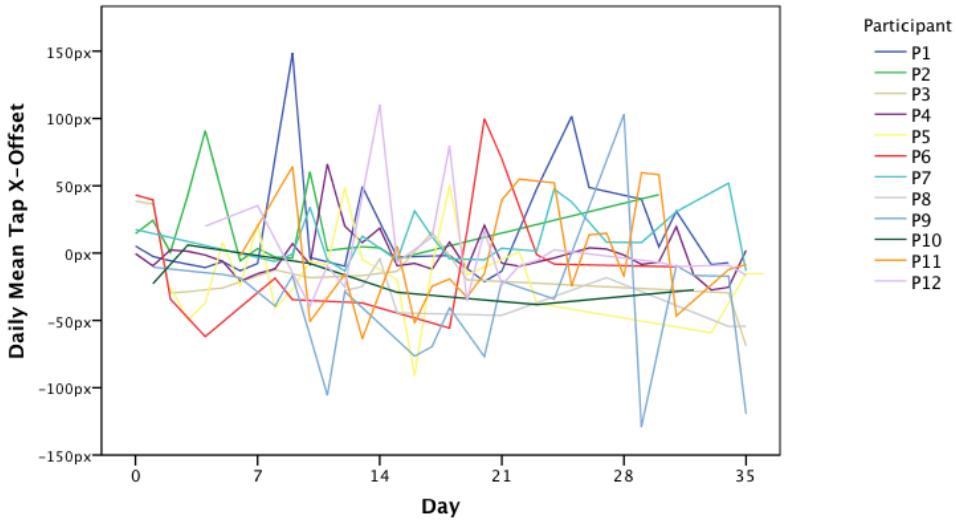

Figure 7.8 Line graph showing the daily average x-offset of each participant's tap gesture behaviours. Values <0px are left of the target centre, values >0px are right of the target centre.

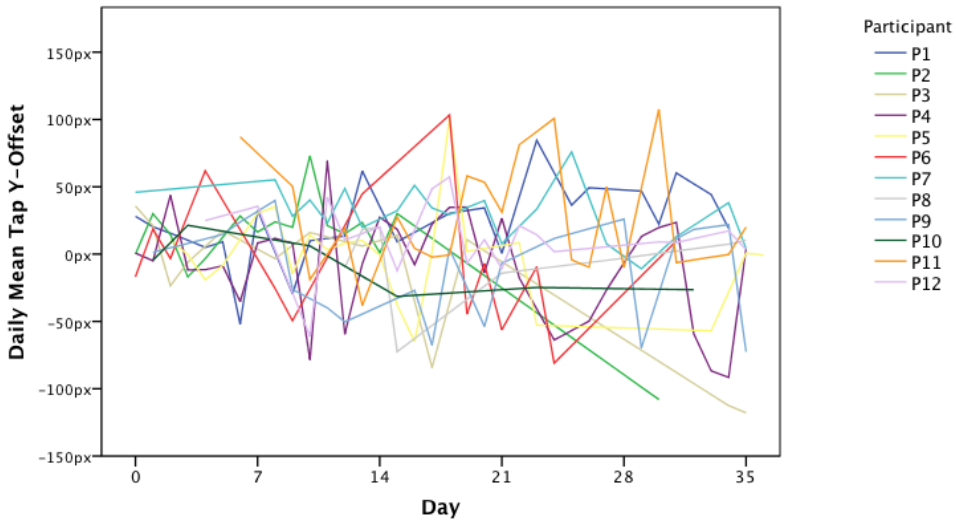

Figure 7.9 Line graph showing the daily average y-offset of each participant's tap gesture behaviours. Values <0px are above the target centre, values >0px are below the target centre

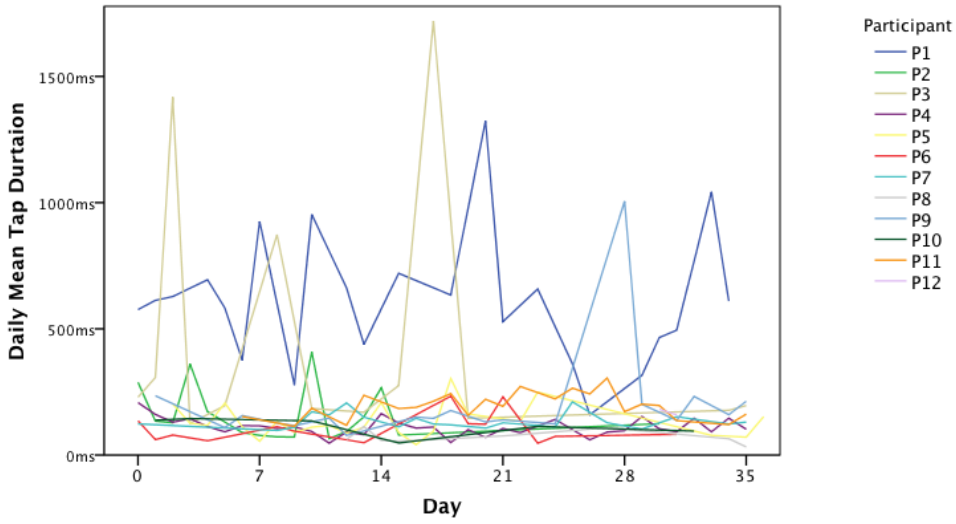

Figure 7.10 Line graph showing the daily average duration (milliseconds) of each participant's tap gesture throughout the in-situ study.

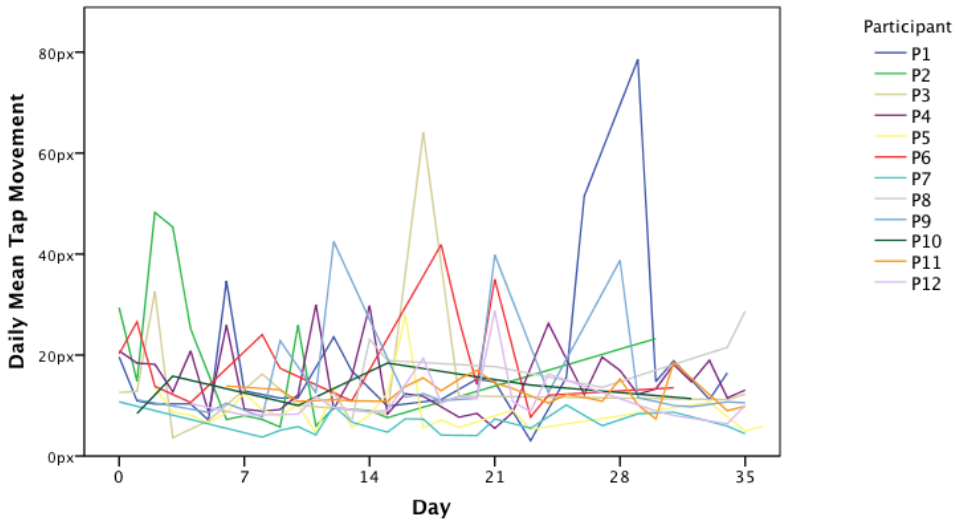

Figure 7.11 Line graph showing the daily average movement (pixels) within a tap gesture for each participant throughout the in-situ study.

7.3.2.2 Usage Behaviours

These results suggest that not only are the interaction behaviours significantly different between participants, but in most cases they are also significantly different

between sessions for the same user. Specifically, this suggests that interfaces need to be able to respond and adjust to changes between interaction sessions for each user.

The flexible schedule design of the current study meant that participants were free to completely immerse themselves in the device for hours on end, or forgo using the devices for a number of days at a time and completely avoid interactions with an application, which was observed for P1, who had very few sessions with the memo application throughout the study. At the same time, participant P4 showed almost obsessive interactions behaviour, playing games of Sudoku for long periods of time, late at night and into the early hours of the morning. Figure 7.12 illustrates the number of gesture instances collected by each application for the participants throughout the evaluation.

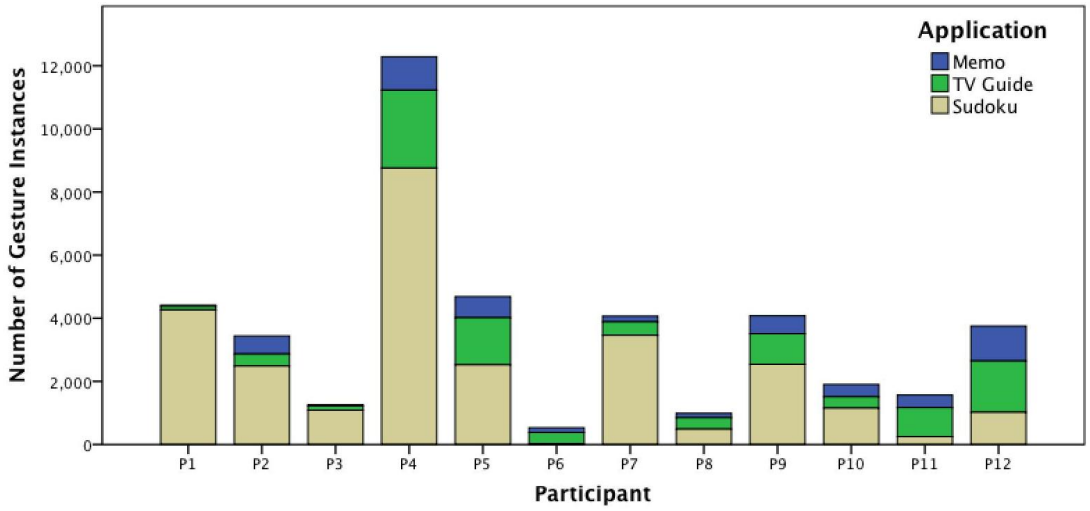

Figure 7.12 Number of gesture instances captured from each participant per application

The decision to give participants the power to dictate their own interaction schedules came from a desire to understand the natural usage patterns and interaction habits of

the participants in their everyday lives. This decision has yielded valuable insights into the highly variable usage behaviours within and between participants, and these were tested for homogeneity of variance, unequal variances were observed as ($p<.001$). Specifically, these results suggest that usage of the devices did not follow a particular schedule or pattern, but rather the participants interacted with the devices in an informal, unstructured manner. Furthermore, the usage behaviours between participants were significantly different. Figure 7.13 illustrates the individual participants' usage share for each application throughout the evaluation.

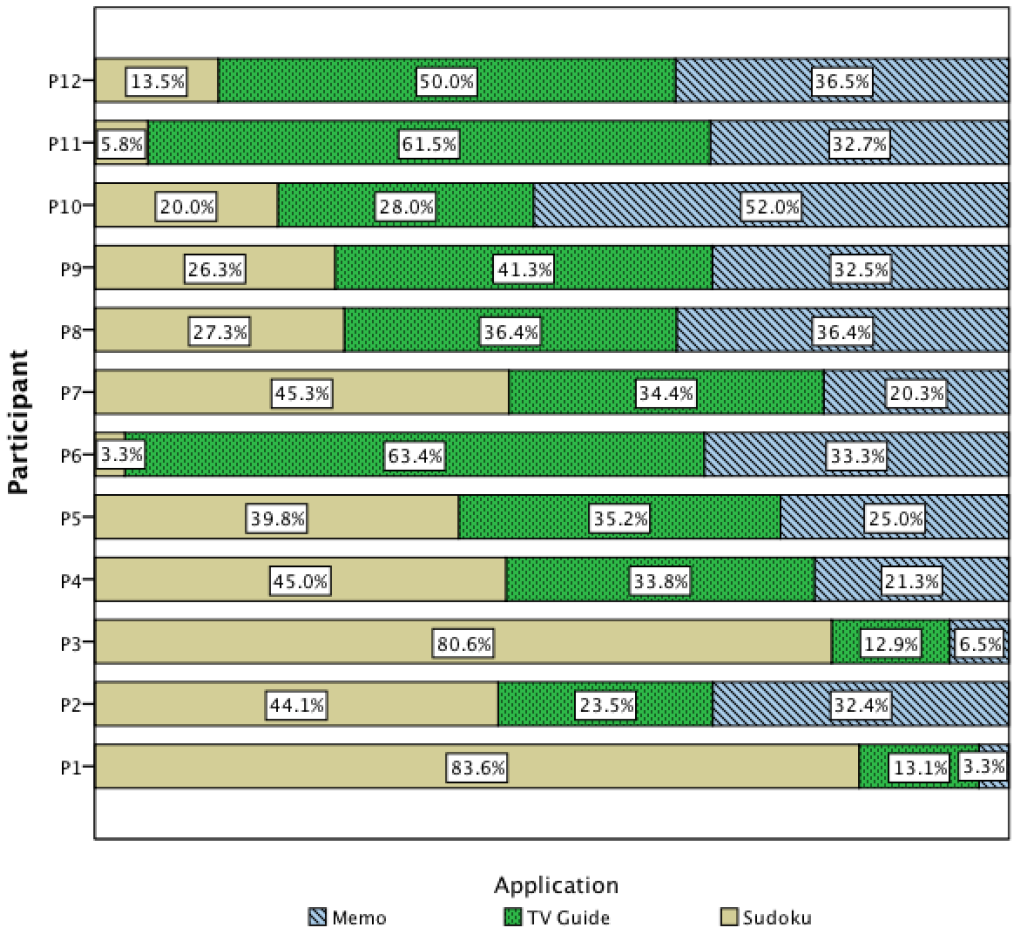

Figure 7.13 Application usage share for each participant.

7.3.3 General Interaction Behaviour

While participants displayed very personal and diverse usage patterns and behaviours, the study also uncovered more generic shared behaviours across the participants. This next section will discuss the shared interaction behaviours in relation to gestures both dependent on and independent of interface components.

7.3.3.1 Unique Features between Components

The data collection process supported by the SUM framework relates touch interactions to iOS interface components such as UITableView's (i.e. Figure 7.1 and Figure 7.2) and UIButton's (i.e. Figure 7.1, Figure 7.2 and Figure 7.3), allowing the data to be analysed and presented in relation to these interface components, rather than the individual applications or pages. Table 7.3 provides an overview of the UI component classifications applied for this analysis.

UI Component	Group Classification	Gesture Recognisers
Table view	List	Tap & Swipe
Date Picker		
Button (width < 50%, and height < 20% of the screen)	Button	Tap
Number Pad		
Switch		
Button (width greater than 50% of screen)	Wide Button	Tap
Text views		
Navigation Bar	Non Interactive	None

Table 7.3 UI component classification into lists, buttons, wide buttons and non interactive.

All of the experimental applications made use of the iOS UITableView controls to create lists of menu items and application content (e.g. TV programme listings Figure 7.3, or Today's Memo list Figure 7.1). TableViews or lists are highly

common application interface components within both the stock and third party mobile applications for all of the major mobile OS platforms (including iOS).

The default configuration of the TableView control creates a large interface component typically spanning the full width and height of the mobile display, responding to both vertical swipe and single tap gestures. The lists are commonly populated with left-aligned headers, long text descriptions, and/or images. Each row is separated with horizontal borders on the top and bottom. All of the lists used within the experimental applications contained either header text only, or header text with subtext details (Figure 7.14), all of which were text aligned to the left.

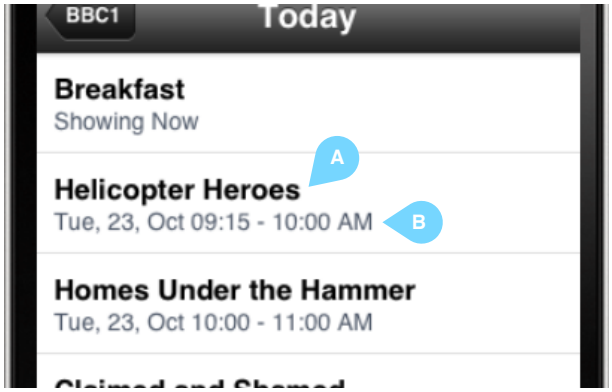

Figure 7.14 TV Guide today list, showing the item header (A) and subtext detail (B).

A Kruskal-Wallis test was run to determine if there were differences in the horizontal location of taps when interacting with list control elements, as illustrated in Figure 7.14. Statistically significant differences were identified between the numbers of taps across the horizontal locations, $\chi^2(9)=35.362$, $p<.001$. Similarly, a Kruskal-Wallis test was run on the vertical location of taps within list control elements which revealed a statistically significant difference across the vertical locations, $\chi^2(14)=171.713$, $p<.001$. While the configuration of the lists allowed users

to select items by performing a one-finger single tap gesture anywhere along the horizontal space of the list item, the analysis of tap gestures with list items suggests that participants performed the tap gestures within the left side of the display as shown in the touch heat map in Figure 7.15. Likewise, a greater number of tap gestures occurred in the upper vertical screen locations and the lower locations.

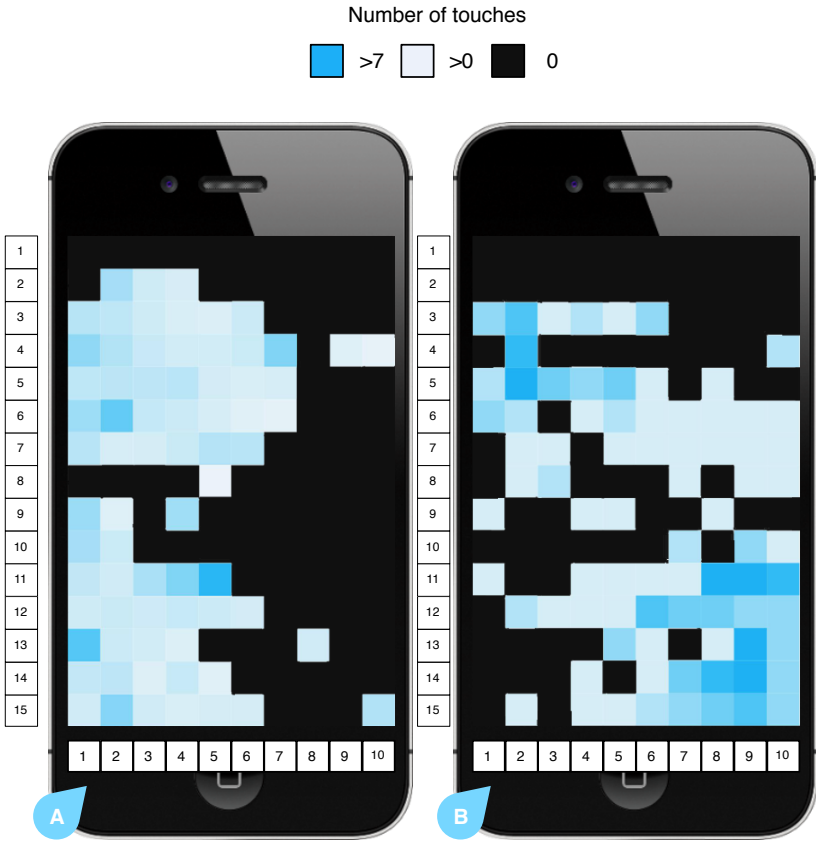

Figure 7.15. Heat map of tap gesture selections (A) and the origin of swipe gestures (B) within onscreen targets spanning the full width of the screen.

These results echo a similar behaviour that was observed in the laboratory study (Chapter 5) with other applications using list controls, that participants were wrapping their left-hand fingers around the device, with their thumb on the left and their fingers on the right when performing a vertical swipe gesture. When asked

about this behaviour, the participants reported this placement was an attempt to provide additional support when performing the swipe gestures. However, as a result of this gripping technique the participants created a number of unintentional item selections.

While the heat map swipe gesture locations may appear to be located around an anchored thumb pivoting around the bottom right of the device, all of the participants within the in-situ study reported either resting the device on a flat surface or holding it in their left hand and interacting with the device using their right hand. This suggests that the tendency to interact with the left-hand side of the lists is a conscious decision, opposed to the limited reach of their thumb or finger. One possible reason for this behaviour is the alignment and position of the text content along the left side within the list items, providing the users with a clear visual target to interact with.

A Kruskal-Wallis test was run to determine if there were differences in tap duration between the horizontal and vertical locations of list control elements. Tap duration was statistically significantly different between the horizontal locations (as labelled in Figure 7.15), $\chi^2(9)=70.031$, $p<.001$. Post-hoc analysis revealed statistically significant differences in tap duration (in seconds) between horizontal locations 1 (*Mdn*=.19) and 2 (*Mdn*=.15) when compared with locations 3 (*Mdn*=.14)(*p*=.002)(*p*=.004), 4 (*Mdn*=.13)(*p*<.001), 5 (*Mdn*=.13)(*p*<.001) and 6 (*Mdn*=.12)(*p*=.003)(*p*=.011). Specifically, these results suggest that tap durations are significantly shorter when interacting with list elements in horizontal screen locations further away from the edges of the screen. However, no statistically

significant differences were identified in tap duration between the vertical locations, $\chi^2(14)=22.322$, $p=.051$.

Figure 7.5 also illustrates the origin of swipe gestures with list controls within the experimental applications. Kruskal-Wallis tests were run to determine if there were differences in the horizontal and vertical origin locations of swipe gestures within list control elements. Statistically significant differences were observed between the vertical screen locations for the origins of swipe gestures, $\chi^2(13)= 34.987$, $p=.001$. No statistically significant differences were observed between the horizontal screen locations for the origins of swipe gestures, $\chi^2(9)= 8.991$, $p=.438$. However, this data included vertical and horizontal swipe gestures, a Kruskal-Wallis test was run on the vertical swipe gestures only to determine if there were differences in the horizontal locations. When only looking at vertical swipe gestures, statistically significant differences were observed between the horizontal screen locations and the origins of swipe gestures, $\chi^2(9)= 18.513$, $p=.03$. Specifically, the results suggest that when users interact with list control elements, swipe gestures are more likely to originate from the bottom right hand side of the screen as illustrated in Figure 7.15. This location would provide the most optimal start point to vertically scroll through the maximum number of items, while also allowing the participants to view and read the left-aligned textual content of the lists.

Kruskal-Wallis tests were run to determine if there were differences in the duration of swipe gestures between the horizontal and vertical origin locations. No statistically significant differences in swipe duration were observed between the horizontal, $\chi^2(9)= 7.545$, $p=.581$ or vertical origin locations, $\chi^2(13)= 14.506$, $p=.270$.

These results suggest that the timing behaviours of a user's swipe interactions are independent of the spatial location that the swipe originates from. Similarly, no statistically significant differences in the swipe distance were observed between the horizontal, $\chi^2(9)= 7.444$, $p=.591$ or vertical origin locations, $\chi^2(13)= 20.108$, $p=.065$. Therefore, these results suggest that the distance lengths of swipe gestures are independent of the spatial location that the swipe originates from.

A Wilcoxon Signed-Rank test was run to determine if there were differences in the duration of tap gestures vs. swipe gestures. There was a statistically significant increase in the duration when users performed swipe gestures (*Mdn*=.475) compared to tap gestures (*Mdn*=.207), $z =4.339$, $p<.001$. Specifically, these results suggest that the duration feature could be used to aid in the distinction between tap and swipe gestures within list control elements. This is particularly useful in cases where a user's tap gestures contains unintentional movement data, thus causing confusion between the intent of a tap or swipe gesture.

7.3.3.2 Interaction with Buttons

Within this analysis only interactions made with button components as defined within Table 7.3 were included. The target also had to respond to a one-finger tap gesture, but not exclusively the one-finger tap gesture if components outwith the button classification were excluded from the analysis of horizontal touch offsets, since the users could successfully interact with the components anywhere along the horizontal axis. The analysis of tap interactions made with buttons revealed participants' target offset behaviours reflected those collected in similar studies investigating touchscreen performance measurements (Henze et al., 2011; S. Lee &

Zhai, 2009; Y. S. Park & Han, 2010). The device screen was divided up into a 10x14 grid as shown in the heat maps of Figure 7.5, and buttons were grouped into these locations based on their origin point. A Chi-square test for association was conducted between the horizontal touch offset and vertical screen location. There was a statistically significant association between horizontal offset and vertical screen location, $\chi^2(26)=1906.54$, $p<.001$. Table 7.4 summarises the observed results of the horizontal offsets by vertical screen locations.

Vertical Screen Location	Horizontal Offset Location			
	Left	Origin	Right	Total
1	0.0%	0.0%	0.0%	0.0%
2	0.0%	0.0%	0.0%	0.0%
3	0.3%	0.0%	1.6%	2.0%
4	3.8%	0.2%	3.0%	6.9%
5	3.6%	0.3%	5.4%	9.2%
6	4.3%	0.3%	8.9%	13.4%
7	4.0%	0.3%	8.0%	12.3%
8	2.4%	0.2%	6.7%	9.3%
9	1.9%	0.1%	2.1%	4.1%
10	5.1%	0.2%	0.7%	6.0%
11	5.5%	0.3%	1.2%	7.0%
12	7.6%	0.3%	1.1%	9.0%
13	4.8%	0.3%	0.8%	6.0%
14	4.2%	0.1%	0.5%	4.8%
15	8.4%	0.3%	1.3%	10.0%
Total	55.8%	2.8%	41.4%	

Table 7.4 Summary of horizontal touch offset locations across the vertical screen locations. 1 = top edge, 15= bottom edge.

A Chi-square test for association was conducted between the horizontal touch offset and horizontal screen location. There was a statistically significant association

between horizontal offset and horizontal screen location, $\chi^2(18)=827.31$, $p<.001$. Table 7.5 summarises the observed results of the horizontal offsets by horizontal screen locations. Specifically, these results suggest that horizontal touch offsets are influenced by both the vertical and horizontal screen locations of the interaction.

Horizontal Screen Location	Horizontal Offset Location			
	Left	Origin	Right	Total
1	4.6%	0.2%	2.3%	7.1%
2	5.9%	0.5%	3.6%	10.0%
3	3.2%	0.3%	3.8%	7.3%
4	2.5%	0.2%	2.4%	5.1%
5	7.8%	0.2%	3.0%	11.0%
6	2.2%	0.3%	5.2%	7.8%
7	3.9%	0.2%	4.9%	9.0%
8	15.2%	0.3%	4.5%	20.0%
9	9.0%	0.6%	5.9%	15.5%
10	1.6%	0.0%	5.6%	7.2%
Total	55.8%	2.8%	41.4%	

Table 7.5 Summary of horizontal touch offset locations across the horizontal screen locations. 1 = left edge, 10 = right edge

7.4 Discussion

The primary objective of this user evaluation was to obtain accurate measurements of touchscreen interactions from participants within a real-world context to understand how user performance and interaction behaviours fluctuate over a four-week period. Furthermore, the objective was also to evaluate the refinements to the SUM framework, and assess the potential of the SUM framework as a tool to accurately measure user performance from remote real-world interactions.

7.4.1 SUM Framework and Data Collection

Firstly, while the SUM framework successfully captured and returned the user interactions for all 12 participants, for two of the participants substantial amounts of interaction data were lost. Although this loss of data was attributed to human error and a limitation of the device platform, it did expose a large vulnerability within the SUM framework data storage. Updated versions of the SUM framework have however address these issued by storing all captured interactions locally and external from the applications. Therefore, in the event of a participant accidentally removing the experimental application, no further interaction data would be lost.

Secondly, the SUM framework was passively recording user interactions from the three experimental applications. Therefore, it was able to capture information relating to the usage behaviours of the users with the applications, for example only using the memo application on a Monday afternoon. Although these usage patterns were of interest and exposed void periods of no interactions and data collection, the result of this passive approach meant that some participants were much less engaged with the technology. Thus, fewer sessions and interactions were captured for those participants. Future evaluations might consider augmenting the participant driven usage patterns with periodical prompts to engage with the applications, this could be achieved through the notification services that exist on mobile platforms.

7.4.2 Real-World Interaction Behaviour

The experimental applications for this study were selected to represent the real-world uses of mobile touchscreen devices. One participant stated that the applications were not of interest or use within his/her daily life. However the remaining participants

described their experience within the evaluation as simply playing with the device without feeling as though they were being observed or forced to perform tasks. Therefore, the interactions captured within this user evaluation represent the real-world behaviours of the participants. As such, this study has demonstrated that existing approaches to obtain measurements of tremors from device accelerometer motion is not supported by the holding behaviours of participants with hand tremors in a real-world context. Thus, alternative methods leveraging the resulting onscreen interactions need to be explored.

Furthermore, analysis of the touchscreen interaction characteristics identified that participants behaved significantly differently to one another regardless of sharing a stereotypical disability classification, suggesting that interface adaptations should be defined based on an individual's abilities and not their disability. However, it was also observed that an individual's interaction characteristics fluctuated significantly between application sessions. Therefore, it is not enough to use an individual's data to train the device's input gesture recognisers, adaptations need to be made based on the individual's current abilities and behaviours. Thus, it is essential for interface adaptations to factor interaction context, and they should be applied on a per session basis.

7.5 Conclusions

This chapter reported on a four week in-situ user evaluation that investigated the real-world interaction characteristics and usage behaviours of users with visual and motor impairments. The evaluation explored the use of the revised SUM framework (Chapter 6) as a method of collecting measurements of individuals' interaction

abilities from real-world interactions with the three experimental applications: Memo; TV Guide and Sudoku, across a four week period. Analysis of participants' interactions demonstrated that interaction characteristics differ significantly both between participants and between sessions of the same participant. However, the current input gesture recognisers of mobile touchscreen devices require that participants be able to perform gestural actions consistently with the device's or application's predefined recogniser parameters. Based on these results it would be concluded that individuals with fluctuating abilities could benefit from input gesture recognisers that can accommodate these variances in performance to improve the recognition accuracy.

Chapter 8. Applying Context to SUM

After conducting a in-situ user study to capture the real-world interactions of mobile touchscreen users (Chapter 7), this chapter now explores the effect of training user models from this data, and using the resulting models to perform interface adaptations on the touch gesture recognisers.

8.1 The Need for Contextual Measurements

Previous works have proposed interface solutions targeted to a particular user group (Guerreiro, Nicolau, Jorge, & Gonçalves, 2010a; Hurst, Hudson, Mankoff, & Trewin, 2008a; Nicolau & Jorge, 2012b; Trewin et al., 2006; Wacharamanotham et al., 2011) while others have applied adaptations to create personalised interfaces for a specific user (Findlater & Wobbrock, 2012; Flatla & Gutwin, 2011; Gajos et al., 2007; Heron et al., 2013; Trewin, 2004). However, the analysis of the user interactions from the in-situ evaluation demonstrated the significant variances in user behaviours and performance between participants (regardless of belonging to the same stereotypical user group) and between sessions from the same participant. Therefore, the user models need not only to be specific to the individual, but must also adjust to the context or situation for which the interactions take place. Thus, it was concluded that simply training user models based on a user's data would not be enough to address the fluctuations in performance. To mitigate the between session differences of user performance, a novel approach leveraging contextual measurements of interactions to predict session behaviours and needs has been proposed. Contextual measurements were used to refine the training data selection when building shared user models.

8.2 Extracting Features and Intent

In order to create and evaluate the user models, the interaction features needed to be extracted to define the parameters of the new SUM gesture recognisers. The dataset was collected within the in-situ user evaluation, whereby all data was captured from three real-world mobile applications. As a result of using real-world interactions, the user's actions and intentions were unknown for each interaction. Therefore, methods to obtain values for the user's intent for each action needed to be defined to evaluate the accuracy of the models.

8.2.1 Touch Features

This sub-section describes the touch features extracted from the touch gestures captured by the SUM framework.

Touch Location (X, Y): Represents the horizontal and vertical location of the user's finger when it was lifted from the screen. These locations are absolute values measured in relation to the physical screen dimensions.

Touch Offset (X, Y): captures the user's *x* or *y* offset between the touch begin (finger down) and end (lifting the finger off) states.

Touch Duration: captures the time duration between the first and final state of a touch gesture.

Absolute Touch Movement: measures the total Euclidean distance between all of the touch states of a gesture.

Straight-line Touch Movement: measures the Euclidean distance between the first and last touch states of a gesture, the combined touch x and y offset.

Relative Touch Movement: calculated as the ratio of *straight-line movement* to *absolute movement* to measure the amount of additional or unintentional movement within the gesture.

Movement Direction Changes (X, Y Axes): measures the number of direction changes within the chosen axis or the combined horizontal and vertical, collected from the touch movement states.

Target Offset (X, Y): captures the user's *x* or *y* offsets from the centroid location of the target interacted with during the touch gesture.

8.2.2 Extracting Touch Intent

Within controlled laboratory user evaluations it is relatively straightforward to establish a user's intended actions, typically the design of the study is such that users have a clear goal, thus error identification is easy. For example, their brief would be to tap the onscreen targets as quickly as they can with their dominant hand. The resulting dataset would contain user touch information where the intended gesture and target are known. However, when conducting in-situ user studies it is unreasonable to assume that the each user interaction carries intent, or that the device correctly interpreted the user's intent. Therefore it is important to apply methods to discriminate between actions with and without intent. Recently, Gajos et al. (2012) conducted in-situ observations of user performance with computer pointing devices, combining both task specific observations and natural computer interactions. The goal of the research was to develop techniques to discriminate between interactions made with focus and intent, from those made while the users were distracted. Gajos et al. applied Fitts' law models to the user data collected during the pointing tasks to

correlate features of intentional pointing movements, and develop classifiers to identify intentional pointing actions within the natural user interactions. The in-situ evaluation design (Chapter 7) mirrors the approach of Gajos et al. (2012), providing participants with a task for each application to enable the collection of interaction data with intent, while also collecting the natural interactions of the users outwith the application specific tasks. The tasks for each application relate to the example sheets provided to the participants during in-situ study training session (Chapter 7). It was possible to automatically discriminate the Sudoku task data from the normal gameplay data using the unique page identifiers generated when starting a new task game, vs. those from the normal gameplay. Using these unique page identifiers it was then possible to cluster any interactions that were associated with the user performing the tasks with the example sheets. This was not possible from the datasets collected by the TV Guide and Memo applications, as no specific task pages existed within the applications themselves. However, sessions where participants completed tasks within these applications could be identified using the session timestamps and comparing the user's task sheets, which served as an interaction diary.

8.2.2.1 Fitts Modelling

In order to apply the procedures outlined by Gajos et al. (2012) for obtaining in-situ measurements of intent, Fitts models were constructed from the participants' touch data collected during the tasks. Previous attempts to apply Fitts models to touchscreen interactions in the wild demonstrated that the models were not an accurate fit of the interaction data (Henze & Boll, 2011). Using linear regression to determine the intercept and slope of the regression line as required in the Shannon

formulation (MacKenzie, 1992). The line intercepted (a) at .05 with a slope (b) of 1.67 with a resulting correlation of r=.169. Specifically, the low correlation between the touch interactions and the Fitts' models suggests that this approach did not work for the dataset. Therefore, this method was not utilised to classify the user intent of the natural interaction behaviours of the participants, instead methods leveraging the application domain and strategies were investigated.

8.2.2.2 Sudoku Game Modelling

Participants were provided with puzzle solutions for each of the Sudoku tasks within the application, and asked to periodically complete tasks from the list by copying over the puzzle solution values from the sheets into the game. The participants were not required to solve the puzzles but instead perform a data entry task, thus removing the cognitive problem solving factors that would influence their interaction behaviour. Therefore, any deviation from the required task-input was classified as an unintentional interaction. The Sudoku *Task Model* was defined to apply the Sudoku game logic and puzzle solution to identify user intent for interactions. By leveraging the Sudoku game strategy and puzzle solutions, it was possible to predict user intent for interactions and refine the recognised touch gestures to reflect the user intent. Figure 8.1 illustrates a possible scenario, where a user selects the Sudoku cell; the *Task Model* predicts the user's next move to be entering the number 6; the user successfully taps the number 6 button thus allowing intention estimations to be confirmed for the initial cell selection and the entered value.

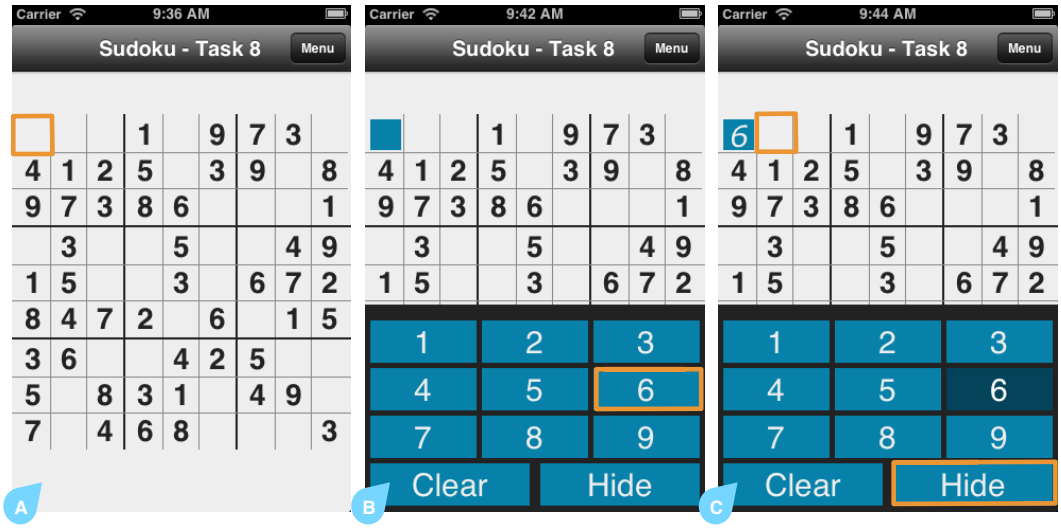

Figure 8.1 Sudoku game modelling interaction predictions with correct targets. (A) Starting board view, (B) user selected cell, predicted next move as number pad button '6', (C) user enters '6' and next predicted moves are adjacent cell or hide button.

Alternatively, the Sudoku *Task Model* can be used to refine tap gestures with the wrong target as shown in Figure 8.2. The scenario is the same as illustrated within Figure 8.1, however the user taps the number 5 rather than the predicted number 6 button; the user's tap gesture is updated to shown the intended target to be the predicted input. Similarly, had the gesture been unrecognised due to exceeding the timing or movement thresholds of a tap gesture, the Sudoku *Task Model* allows the gesture intent to be refined and reclassified as a successful tap gesture.

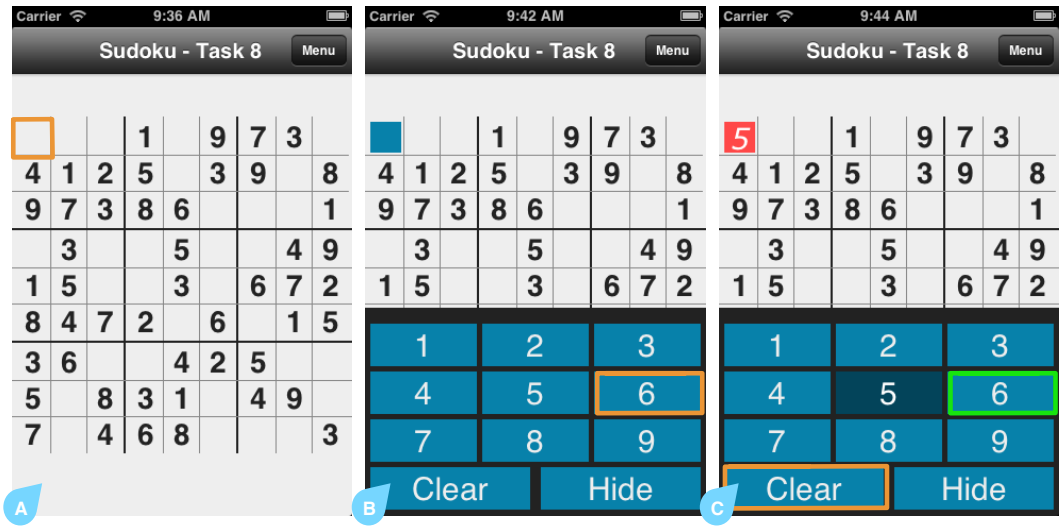

Figure 8.2 Sudoku game modelling interaction predictions with incorrect targets. (A) Starting board view, (B) user selected cell, predicted next move as number pad button '6', (C) user enters '5' the target intent is updated to the number '6' button and the next moves are predicted as '6' or the clear button.

During the in-situ user evaluation participants only completed 46 Sudoku tasks, but played a further 266 games of Sudoku. Despite there being no differences between the Sudoku task and game interfaces, the Sudoku *Task Model* could not be applied to the captured games of Sudoku as it assumed the user knew the correct solution and would not enter incorrect values intentionally. However, this is not always the case when participants are playing games of Sudoku rather than performing the task puzzles, within the Sudoku games participants were required to solve the puzzle themselves, thus, were free to make mistakes and guess cell values that could in fact be the incorrect values. For example, the scenario outlined within Figure 8.2 assumes the user was aiming for the number 6, and tapped the number 5 in error. In a game situation the user could have guessed the number 5 was the correct answer for that cell and therefore touched that target with intent. To prevent the Sudoku *Task Model* from refining the gesture target to being the predicted value of the number 6 an alternative model was defined, the Sudoku *Game Model*.

The *Game Model* did not leverage the puzzle solution to predict the participants next move, it only relied on the Sudoku game logic. Therefore, wouldn't correct gestures where the interactions were intended but simply the wrong values as a result of the user guessing or not knowing the correct answers. The *Game Model* could still refine intent for gestures where the user selected the wrong targets, and where the gesture itself was not recognised due to performance deviation. These scenarios are detailed below.

Wrong Target. One possible scenario for intent correction is when the participant taps a target that does not respond to the tap gesture, implying that a nearby target would have been the intended target.

The *Game Model* captures such scenarios in the following way, illustrated in Figure 8.3. The participant taps the cell containing the number '4', which is not an interactive object and therefore no interaction feedback is provided. However, SUM records the tap gesture marking this object as the target. The participant next taps a nearby empty cell target that is interactive. Potentially this was the intended target for the previous interaction, however there is still uncertainty. The next probable moves are either tapping the 'hide' button to remove the number pad, signifying that the user is happy with their number selection. Alternatively they may tap a new cell and enter the next number. If either of these possible interactions occurs then the game model marks the cell (B) as complete and the number '6' as committed. At which point the intended target for the original tap gesture (A) is refined to the empty cell above (B), moreover the other tap gestures are confirmed as intended tap gestures and targets.

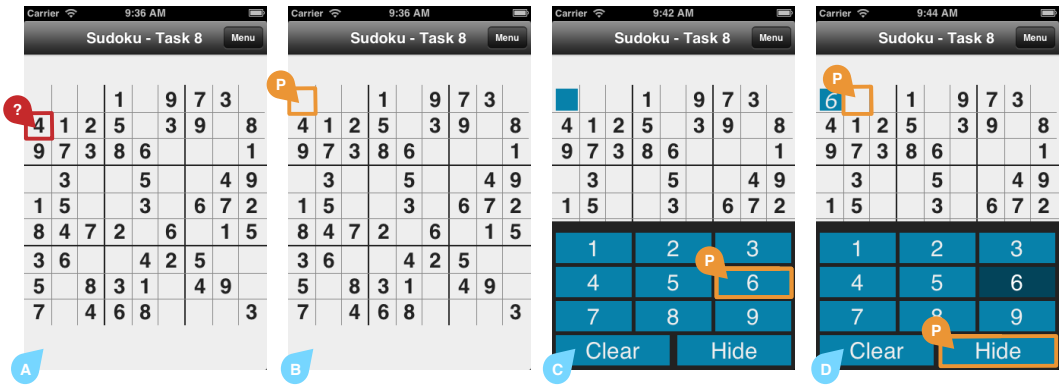

Figure 8.3 Game model refining the target intent for a wrong target error.

Unrecognised gesture. Another common interaction error occurs when the participant performs a gesture that is unrecognised by the device. Possible reasons for gestures being unrecognised is due to timing or movement values outside of the acceptable parameters for the gesture. The following example details how the Sudoku game modeller handles unrecognised gesture errors, to obtain refined estimations of intent, illustrated in Figure 8.4.

The participant attempts to perform a tap gesture in the empty cell (A), but the tap duration exceeds the maximum duration parameter of the gesture recogniser. No interaction feedback is provided to the participant, but the gesture is captured and recorded by the SUM framework as an unrecognised gesture. Next the participant repeats the gesture, this time it is recognised by the device and the cell receives the tap gesture. The scenario then plays out as detailed above for the _Wrong Target_. The Sudoku modeller can then refine the gesture type of the original interaction (A) from being an unrecognised gesture to being an intended tap gesture.

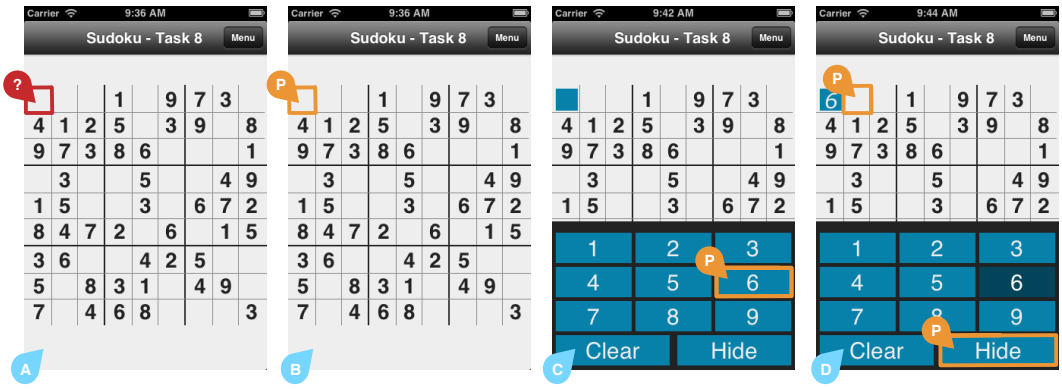

Figure 8.4 Game model refining the gesture type for an unrecognised gesture error.

Unrecognised gesture and wrong target. As the name suggests this error occurs when the participant performs a gesture that is unrecognised and with a target that is not interactive. While it would be possible to use the steps detailed above to attempt to refine and correct the intent for these interactions, it was decided not to infer intend for these interactions due to the compound errors.

Within Sudoku task sessions (46) 82.6% of tap gestures were classified as intentional user interactions using the *Task Model*, compared with 84.2% (kappa=89.6%) using the *Game Model,* the results of the models were statistically similar z =-1.524, p=.127. Kappas greater than 75% show an excellent agreement (Fleiss, Levin, & Paik, 2013), therefore the Kappa score between the *Task* and *Game models* demonstrates that the models share an excellent agreement on the classification of gesture intent. Due to this statistical similarity and agreement between the *Task* and *Game Models,* the *Game Model* classifier was applied to the dataset of 33658 (63% of the full dataset) touch gestures collected from the Sudoku application, and obtained refined intent classifications for 26,563 (79%) of the Sudoku interactions. The classifier made no attempted to infer intent from those user interactions that

occur within the application menu system, as the Sudoku game logic could not be applied. These interactions were excluded from the dataset.

Similar attempts to refine the intent classifications of the Memo and TV Guide applications using the participant task sheets were made. However, satisfactory classification rules could not be defined for the interaction behaviours, as it was not possible to identify the goals of interaction from the real-world use of the applications. Thus, accurate discrimination between intentional and unintentional interactions for these applications could not be obtained. Consequently, both Memo and TV Guide were excluded from the testing of model simulations.

8.2.3 Dataset Summary

Using the three experimental applications combined with the SUM framework, a dataset containing over 931 interaction sessions, consisting of 52,650 touchscreen gesture interactions (taps, swipes and unrecognised gestures) was collected from 12 participants throughout the four week in-situ user study (Chapter 7). Table 8.1 summarises the breakdown of the recorded touch gestures during the user evaluation. 26,563 (50.4% of dataset, taps only) of the tap gestures were assigned user intent and target classifications using the Sudoku game model. These classified instances will be used to test the classification accuracy of the shared user models.

Participant	Taps	Swipes	Unrecognised
P1	4434	56	1767
P2	3455	114	1324
P3	1263	34	526
P4	12352	261	1983
P5	4708	46	352
P6	545	17	94
P7	4102	32	201
P8	996	25	76
P9	4135	45	704
P10	1911	19	311
P11	1577	40	335
P12	3770	283	757
Total	**43248**	**972**	**8430**

Table 8.1 Summary of participant gestures captured during in-situ user evaluation.

Table 8.2 summarises the 26,563 Sudoku gestures with intent measurements, showing the number of gestures that were recognised or unrecognised, and where or not they were associated with the correct target. This summary shows that 1051 (39.9%) of all 2633 unrecognised gestures were in fact intended tap gestures on the correct target. Furthermore, 3276 (13.7%) of successful tap gestures were recognised with the wrong target.

	Unrecognised	Recognised	
Correct	1051	20654	**21705**
Incorrect	1582	3276	**4858**
	2633	**23930**	

Table 8.2 Breakdown of device recognised gestures and the resulting intent measurements.

8.3 Statistical Touch Models

Currently within mobile touchscreen interactions the common method of classifying *tap* gestures is the use of the x, y location (either touch begin, or touch end); and fixed movement threshold (movement between the touch begin and end states). However, the proposed gesture recognisers are not defined by fixed parameter boundaries. Instead, the recognisers use statistical probability models to account for the variations in gesture performance between instances. The gesture recognisers used in this evaluation applied Gaussian functions to define the attributes for gesture classification, as illustrated in Figure 8.5. Gaussian functions allow the gesture recognisers to perform classifications based on probability of an action given a series of parameters, as opposed to relying on definitive parameters. For example, Gaussian functions are capable of resolving common touch offset errors whereby the touch occludes two or more possible targets. The target with the highest probability is suggested as the intended target. Similarly, they can account for variances within user performance, such as timing, rather than using a fixed maximum value to threshold all touches above this. A Gaussian function would simply return a lower probability, if the probability of the interaction were greater than it being an unintended touch then the gesture would be recognised. However, the traditional fixed maximum model would not be recognised.

The shared user models defined the parameters of the tap gesture recogniser using the training data to obtain the mean (μ) and standard deviation (σ) values required by the probability density functions. The SUM recognisers were able to apply the classification features as already used by mainstream tap gesture recognisers. The

features included were: x, y location, duration and movement. In addition the models also parameterised the x, y offset. Typically the x, y offset is handled by touch offset models defined on a per device nature, shifting the user's touch input location by a fixed Euclidean vector. However, previous researchers (Buschek, Rogers, & Murray-Smith, 2013; Henze et al., 2012; Holz & Baudisch, 2011; Rogers, Williamson, Stewart, & Murray-Smith, 2011; Weir, Rogers, Murray-Smith, & Löchtefeld, 2012) have proposed user specific touch offsets models, reporting significant improvements in the precision of touch input. The statistical touch models within this evaluation used a threshold of 3σ of each parameter to define success criteria as used previously within the laboratory evaluation (Chapter 5). This threshold was used to aid the distinction between intentional and unintentional touch interactions, whilst allowing for performance variations within gestures.

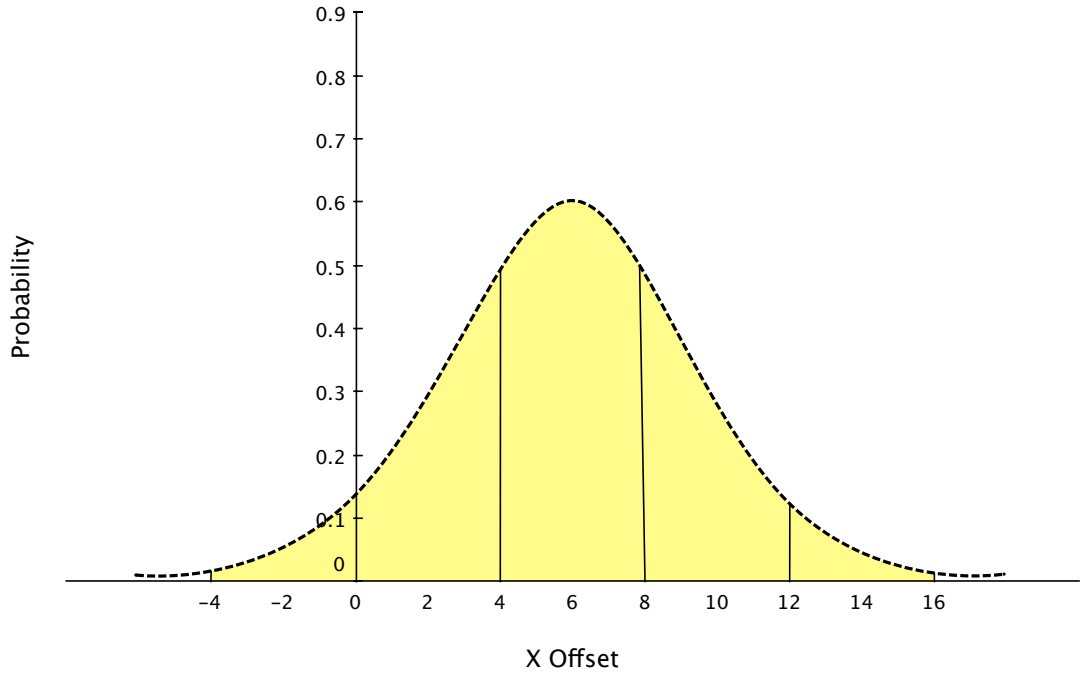

Figure 8.5 Probability density function of tap gesture XOffset

8.4 Refining SUM through Contextual Modelling

Context models are used to define the contributing factors of the user's interactions with the system. Typically these would include the human factors of the user coupled with those imposed by the device and the environment. Previous works have explored systems capable of generating user interfaces based on all of these factors (Macik, 2012). Macik (2012) evaluated the effects of providing interface adaptations that were sympathetic to not only the user model but the model of the device and environment. These adaptations included scaling the font and target sizes, element spacing and line width of the user interface. However, while the approach proposed by Macik (2012) applied visual adaptations based on the context models, the methods used to obtain this information relied on the users manually configuring preferences within a single environment and evaluation instance. As was demonstrated previously within the in-situ user evaluation (Chapter 7), users' abilities fluctuate from session to session. While these users all qualified as having a visual or motor impairment, able-bodied users can equally experience interaction challenges as a results of situationally-induced impairments and disabilities (Sears & Young, 2002). Therefore, manual methods of defining contextual factors and user abilities are undesirable. Furthermore, this approach distinguishes the user factors from the device and environment factors, failing to consider that there might be overlap between the resulting interaction characteristics of a user with hand tremors and an able-bodied user riding a bus. Therefore, the approach proposed by this dissertation was not to use these individual context factors for classification, but rather to work backwards from the interaction characteristics to train models that

respond to these behaviours independent of context factors (user, device or environment model).

There were two key components required to support contextual searching based on interaction characteristics; a method for classifying interaction data independent of the context factors, and techniques to obtain contextual measurements of the user's current interaction characteristics. Furthermore the outstanding questions pertaining to applying contextual measurements to shared user models were:

1. What sample size needs to be measured to obtain accurate contextual features?
2. How much training data is required to build a contextual shared user model?

To answer these questions an independent evaluation of the contextual measurements was conducted. Firstly, this section details the contextual session features used to classify user interactions based on the individual sessions. Secondly, the contextual measurements and distance function used for searching the dataset are described. Then the section reports the evaluation and results. Finally, a short discussion of the findings is presented.

8.4.1 Contextual Session Features

Session features based on the *touch features* (Section 8.4.1) were selected to represent the session, in order to measure the variances of user interaction across sessions and applications and cluster the individual sessions. To ensure the sessions were clustered independent of any stereotypical groups or applications, the features were selected based on the low-level touch interactions. This decision is key to the

sharing of data not only between applications but also between users. The session features represent the average instance and variation of the touch features within the complete session. Each session feature captures both the mean and standard deviation of the corresponding *touch feature.* Similarly the session features are grouped in relation to the gesture type they represent, for example. *Tap* or *Swipe*. Each gesture follows a similar structure of features:

Gesture Duration: measured as the mean and standard deviation duration of all successfully recognised gestures of this type (e.g. tap gestures).

Gesture Offset (X, Y): each axis is measured independently as the mean and standard deviation of the *touch offset* for all the recognised gestures of this type.

Gesture Movement (Absolute, Straight line and Relative): each movement value is measured independently based on the *absolute touch movement, straight-line touch movement and relative touch movement.*

Gesture Target Offset (X, Y): each axis is measured independently as the mean and standard deviation of the *target offset* for all the recognised gestures of this type.

The contextual session features were then normalised using the standard score formula to aid the distance measurement process. Within this evaluation no new interaction sessions were added. Therefore, the population size was known ahead of time, allowing the mean and standard deviation values to be calculated based on the entire dataset. However, in a real world setting, new sessions would continually be added to the dataset. Thus, these values would need to be estimated using random samples.

$$z = \frac{x - \mu}{\sigma}$$

Equation 8.1 Standard score formula, used to normalise the session features within the dataset.

8.4.2 Contextual Measurements

This section discusses the technical approach used to obtain the contextual measurements from the session interactions, firstly, exploring sampling techniques to capture contextual measurements using a subset of the session's interactions to predict the complete session features. Secondly, this section describes the distance method used to compute the similarity of sessions based on the session features.

8.4.2.1 Sampling Windows

In order to train the user models using data that match the current contextual measurements the system must sample the current session's interactions and find similar instances based on the session features. To evaluate the accuracy of measuring the current context of interaction, two types of sampling window were selected: time based, using the duration of the session; and instance based, using the number of recorded touch interactions. As well as analysing these two windowing methods the window sizes were increased to evaluate the performance of each combination.

The accuracy of the windowing method and size were calculated based on the similarity of the windowed session features against the features of the complete session. Feature similarity was measured using Equation 8.2 below, where F_{w_n}

represents the feature measurement for window size *n*, and F_s represents the feature measurement of the complete session.

$$f(w_n) = \frac{F_{w_n}}{F_s}$$

Equation 8.2 Feature similarity measurement – w_n, window of size *n*; *Fs*, feature measurement of the complete session; Fw_n, feature measurement for window size n.

Due to the variable nature of both the length of sessions and number of gestural instances within each session, the window sizes were not defined as absolute values. Instead the window sizes used were percentage shares of the complete session length and number of gestures (i.e. 5,10,15,20...95%).

8.4.2.2 Time and Instance-Based Windows

The sessions were sampled using time measurements (*Time-based*) and gesture instances (*Instance-based*) to create a series of window sizes. Due to the flexible nature of the user interactions, application sessions could span any length of time and consist of highly variable numbers of gesture instances. Therefore, the window sizes were calculated relative to the size of the complete session (duration and gesture count) and sample sizes were defined as the proportion of the complete session. There was a statistically significant increase in measurement accuracy when using Instance-Based (*Mdn* = 90.8%) compared to the Time-Based windows (*Mdn* = 83.5%), z =-61.11, $p < .001$. These results suggest that the type of window sampling has an effect on the contextual measurements of user sessions. Specifically, the results imply that Instance-Based window samples improve the accuracy of the

contextual measurements, as illustrated in Figure 8.6. The accuracy measurements of the Instance-Based windows were compared, and a statistically significant difference between the Instance-Based window sizes was found, $\chi^2(8) = 8197.96, p < .001$. Post-hoc analysis revealed statistically significant differences in contextual measurement accuracy between each of the window sizes, excluding their immediate neighbours i.e. no significant differences between sizes .10 and .05 or .15, however significant differences between .10 and windows sizes greater than .15 ($p<.001$). Specifically, these results suggest that larger sampling windows will produce more accurate measurement predictions. Therefore, the recommended window size should be as large as possible. Window sizes with a share of .20 or more achieved contextual measurement accuracies greater than 70% as illustrated in Figure 8.6.

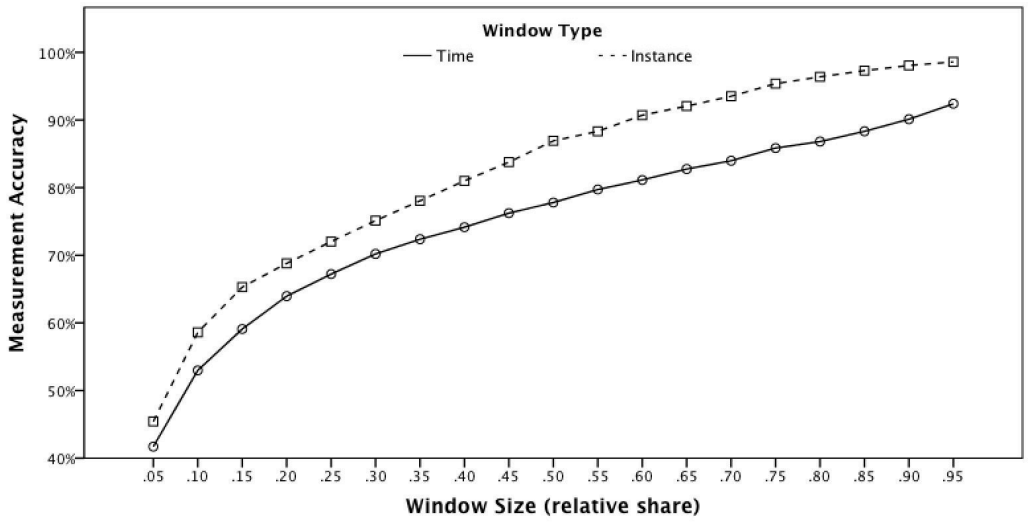

Figure 8.6 Contextual measurement accuracy for Time and Instance-Based window samples.

8.4.2.3 Distance Measurements

The session distances are calculated as the sum of the Euclidean distances of each normalised session feature weighted by the confidence of the session measurements. The distances calculation is shown in Equation 8.3.

$$d = (1 - \alpha) \sum_{i=0}^{n} |F_i^x - F_i^s|$$

Equation 8.3 Session distance formula; where F^X_i, represents the feature value of the current session, and F^S_i is the feature value of the compression session.

Where α represents the confidence of the session measurements for session *S*, given by Equation 8.4.

$$\alpha = \frac{w_x}{w_{\rightleftharpoons}}$$

Equation 8.4 Confidence of session measurements, calculated as the window size of session *x* divided by the optimal window size.

8.4.2.4 Contextual Training Data

A Kruskal-Wallis test was run to determine if there were differences in the gesture recogniser accuracy between the training data sizes of the contextual models. The normalised gesture recogniser accuracy increased by n=50 (*Mdn*=8%), n=100 (*Mdn*=13%), n=200 (*Mdn*=13%), n=300 (*Mdn*=17%) then levelled off until n=900 (*Mdn*=18%), n=1000 (*Mdn*=16%), as illustrated in Figure 8.7. No statistically significant differences were observed between the contextual model training data sizes, $\chi^2(10) = 10.265$, $p = .418$. Specifically, these results suggest that a contextual

model could be trained using n=50 instances and obtain a statistically similar accuracy to a model with n=1000 training instances.

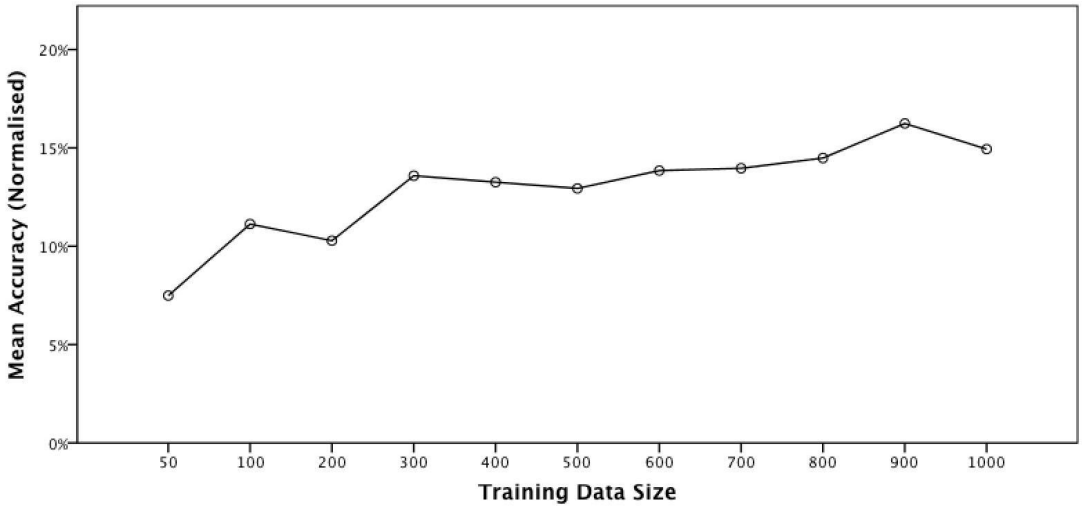

Figure 8.7 Gesture recogniser accuracy improvements with Contextual Models

8.4.3 Contextual Model Summary

This section has proposed and evaluated a novel approach using contextual measurements to source training data specific to the interaction needs and abilities of individual sessions. Following this evaluation is it now possible to answer the related research questions:

1. *What sample size needs to be measured to obtain accurate contextual features?*

Firstly, the evaluation investigated the use of two types of sampling methods, and demonstrated that *Instance-based* windows performed significantly better than *Time-based* sample windows. The *Time-based* windows produced highly variable accuracy measurements as a result of the variable number of instances available at the time of the samples. For example, with a sample size of 5% (of the total session duration) in

one session there were 10 touch instances, while another had only two. However, when using the *Instance-based* window method the accuracy of the measurements was more consistent. Statistically significant differences were found between the window sizes of the *Instance-based* sampling method, the results demonstrated that the longer the sample window the more accurate the contextual measurement predictions were. Although no optimal size exists, it would be recommended that the minimum sample size be 20% to achieve contextual measurement accuracies of 70% and greater.

2. How much training data is required to build a contextual shared user model?

No statistically significant differences were identified in the normalised gesture recogniser accuracy between the training data sizes tested within the evaluation of the contextual models between sizes n=50 (recogniser accuracy improvement of 8%) to n=1000 (16%). Therefore, the contextual shared user models could be trained with n=50 instances of the gesture. Although no significant differences were identified, n=300 (17%) produced a local maximum and its accuracy was not exceeded until n=900 (18%). Therefore, n=300 provides the greatest return and is the recommended training data size.

8.5 Evaluation of Shared User Models

The purpose of this evaluation is to simulate the effects of using the shared user models to tailor the tap gesture recognisers of the Sudoku gameplay. Using the interaction data collected within the in-situ user study as the training and testing data for the shared user models, it is possible to simulate the behaviour of the tap gesture recogniser and measure the classification accuracy against the extracted user

intention values. The simulation will explore the effect of using data from a single user, his/her stereotypical user group and a canonical user model. Furthermore, simulations of models using data from the Sudoku application only, and sharing data between the three applications will also be conducted. The goals of this evaluation were to identify whether the shared user models provide an accurate representation of the individual's needs and abilities; to explore the effects of training models with data from other users, and applications; and finally, measure the effects of applying contextual measurements to refine the shared user models.

8.5.1 Research Questions

This evaluation aims to answer the following research questions:

1. *Can shared user models improve touch recognition accuracy over the default gesture recognisers of the devices?*

2. *Can we use data from other people (within, and outwith the same stereotypical group) to build shared user models that are more accurate than using an individual's own data?*

3. *Can we use data from other applications to build shared user models that are more accurate than using an application's own data?*

4. *Can contextual models improve the accuracy of the shared user models over the Unweighted models?*

8.5.2 Procedure

This section describes the contributing factors of the user models, and outlines the procedure applied to evaluate the performance of each model combination. The three components that define the models are *subject of dataset, domain of dataset* and *selection method.* Figure 8.8 illustrates the 12 component combinations used to create the user models evaluated within this chapter.

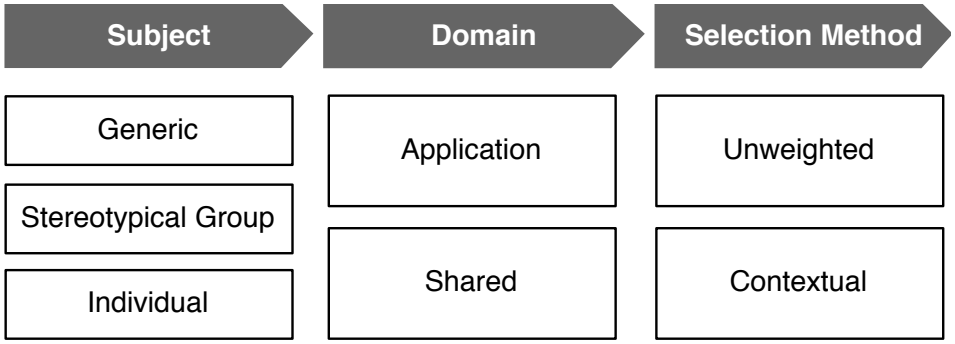

Figure 8.8 Overview of model structures within this evaluation. The three specific components include subject and domain of training data and the data selection method

8.5.2.1 Subject of Dataset

To investigate the effect of creating user models from other participants' interactions the following three *Subject* conditions were defined:

Generic: the touch models were trained using everyone else's interactions, i.e. the current user's data is excluded from the training dataset.

Stereotypical Group: Touch models were trained using interactions from other users that belong to the same group, excluding the current user's data. Participants self-classified during the initial interview stage of the in-situ user evaluation (Chapter 7).

Individual: touch models were trained using only data collected from the participant being tested. Interaction data being tested was excluded from the available training data, and cross-validation with 30-folds was used for each evaluation.

8.5.2.2 Domain of Dataset

To evaluate the effect of creating user models using data from other application interactions the following two *Domain* conditions were defined:

Shared: touch models were trained using data captured within any of the three experimental applications: creating device touch models that are application independent.

Application: touch models were trained using only data captured within the specific application being tested. In a similar way to the *individual* condition from the *Subject* of dataset, this condition ensured training data was not used for testing through cross-validation with 30-folds. This condition represented individual models for each application, whilst maintaining the use of abstract interaction data.

8.5.2.3 Selection Method

To investigate the effect of creating situation-specific models the following two data selection conditions were defined:

Unweighted: which performed a simple randomisation on the available dataset and selected the required number of training instances. Each gesture is treated as an individual instance; therefore there is no guarantee that the training data will come from the same interaction session.

Contextual: which sampled the current interaction session to capture measurements of the user's interaction behaviour. These same measurements were applied to the complete dataset of user sessions, allowing distances to be computed between the current session and each session available within the dataset. The sessions were weighted by their distance from the current interactions, data with the lowest distance most closely matched the current context of interaction. Therefore, this data was selected to train the user models. The distance measurements applied are presented in Section 8.4.2.

8.5.2.4 Training and Testing Data

To perform the simulation and evaluations of the user model combinations, the following dataset conditions were defined:

Testing Data: each simulation required 200 tap gesture instances with intent measurements. These tap gestures were sourced from the user's touchscreen interactions within the Sudoku application. Tap gestures were selected randomly any of the user's Sudoku sessions whereby the gestures had an associated intent measurement.

Training Data: depending on the *selection method* of the current user model condition, training data was defined as 300 tap gesture instances selected at random from all available sessions (in the case of *Unweighted* selection), or from a subset of sessions with similar interaction behaviours (in the case of *Contextual* selection). To ensure that the training data used to build the touch models was also not being used to evaluate the model's accuracy, the 200 testing instances were excluded from the available dataset for training data.

8.5.2.5 Validation of User Models

The goal of this evaluation was to measure the effects of applying user and situation-specific touch models to touchscreen input gesture recognisers, Figure 8.9 illustrates the evaluation process. Baseline performance scores were obtained for the device default configuration by measuring the number of recognised user interactions that match the previously extracted touch intent values. Each model was then scored against these baseline measurements, values greater than zero determined that user models correctly recognised more instances of user intent. In order to reduce the variability of the user model performance measurements, 30-fold cross-validation was applied to each model evaluation, the simulation process is illustrated within Figure 8.11. Sessions were excluded if fewer than 10 touchscreen interactions were captured. Likewise, any model dataset that did not meet the required number of training (n=300) and testing (n=200) instances was excluded from the evaluation, Figure 8.10 illustrates the processes applied to obtain the necessary training and testing data for the simulations. As a result, data from participant P6 was not included in the evaluations due to a lack of testable data with measurements of intent since the test dataset only contained refined classifications of intent for the Sudoku game. Therefore, both the TV Guide and Memo data were excluded from the test data, as no classifications of user intent could be extracted for those interactions. The result of these exclusions produced a dataset containing over 287 interaction sessions, consisting of 33,643 touchscreen gestures available for training models, of which 26,563 gestures contained user intent classifications for testing.

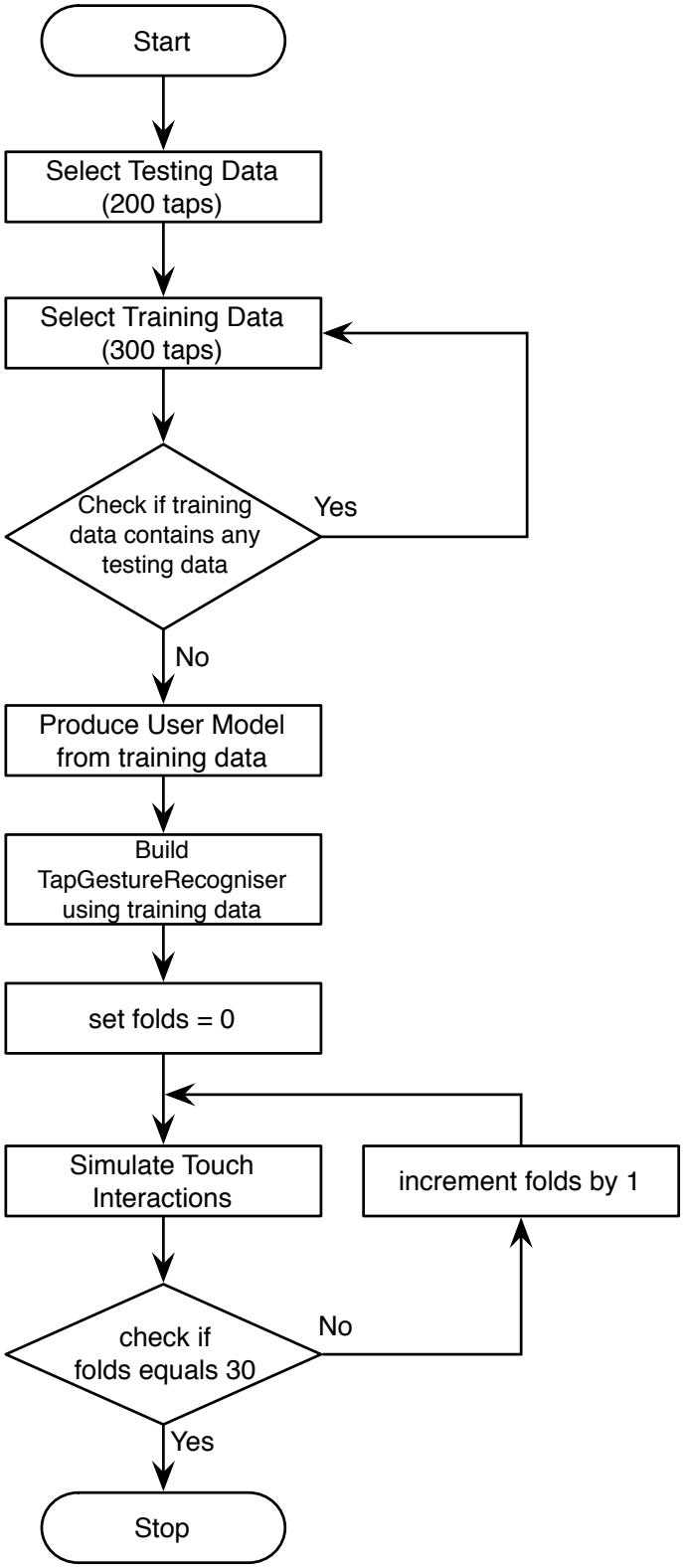

Figure 8.9 Flow chart of the overall evaluation process applied to each user model simulation

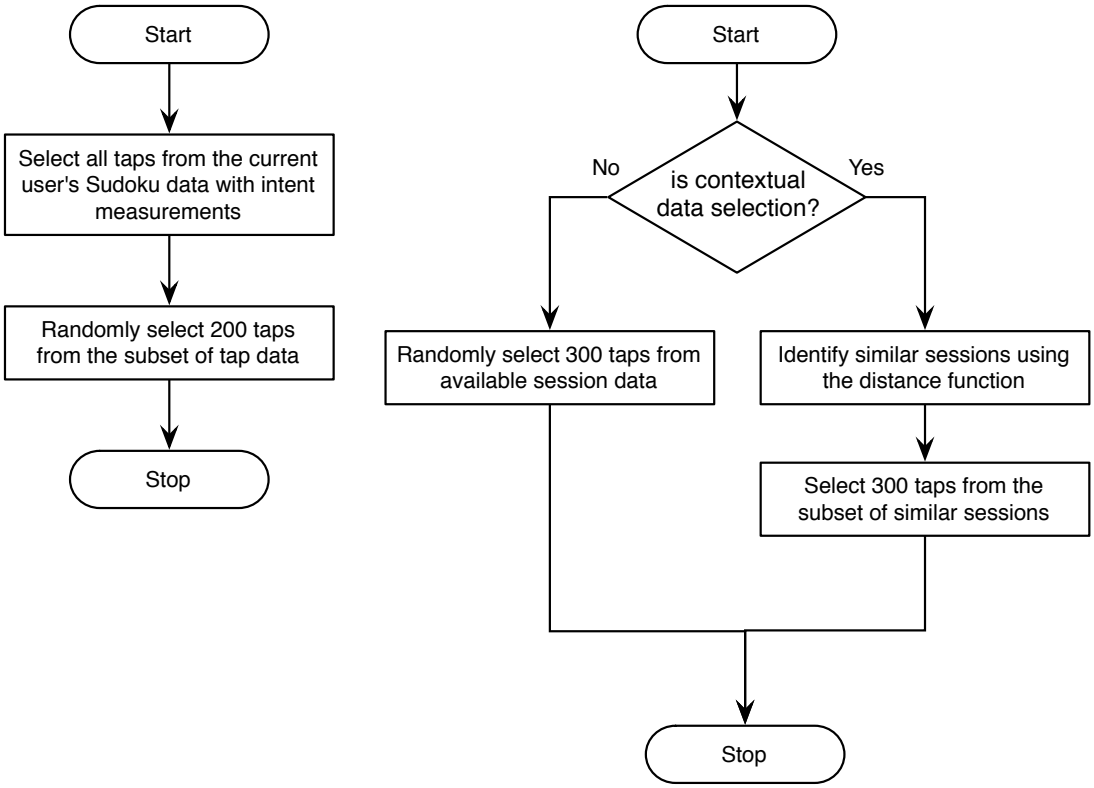

Figure 8.10 Flow charts illustrating the process applied to obtain testing data for the simulations (left) and training data to build the user models and tap gesture recognisers (right).

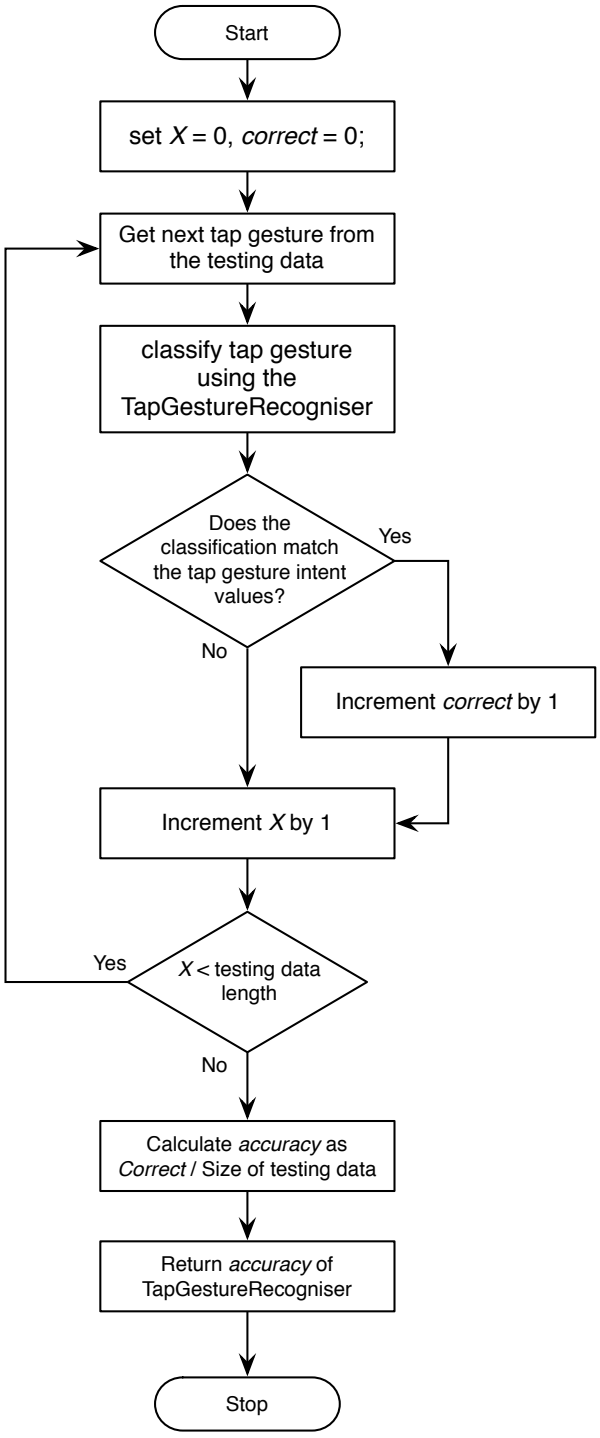

Figure 8.11 Flow chart illustrating the simulation process used to measure the accuracy of the user model conditions and tap gesture recognisers with the testing data.

8.6 Results

This evaluation examined the effects of the subject, domain and selection method conditions. The observed values in all dependent variables were tested using the Shapiro-Wilk normality test. However, the data did not show a normal distribution. Therefore, non-parametric tests (Friedman, Kruskal-Wallis and Wilcoxon) were selected and Bonferroni corrections for post-hoc tests were used. Alpha levels were set as $p < .05$ for significance.

8.6.1 Training and Testing Data

A Kruskal-Wallis test was run to determine if there were significant differences in the accuracy of the *Contextual models* between the training data sizes; no significant differences were found. However, when testing the *Unweighted user models* the accuracy was statistically significantly different between the training data sizes, $\chi^2(10)=20.75$, $p<.023$. Pairwise comparisons were performed with a Bonferroni correction ($p < .00076$) for multiple comparisons. *Post-hoc* analysis revealed no statistically significant difference between user models with a training data size of 1000 and 300 ($p = .735$). Specifically, these results support the decision to train the models using $n=300$ gesture instances. Therefore, each user model was trained using n=300 gesture instances based on these results and the earlier evaluations of the *Contextual models*.

Each measurement required $n=200$ testable gesture instances with valid intent classifications. Therefore, the test data was restricted to interactions from the Sudoku application where user intent could be extracted. A single test consisted of 200 gesture interactions from the specific user; the data was selected at random from all

available test data for that user. 30-fold cross-validation was used to ensure validity of reported measurements; test data was excluded from the training data for each iteration.

8.6.2 User Model Accuracy

The contextual search relied on accurate measurement and prediction of the user's current situation and behaviour. Any inaccuracies or errors in the prediction would have an effect on the accuracy measurements of the user models. Therefore, to ensure that the evaluation of the *Contextual models* only tested the resulting models and not the ability to predict the context of interaction, each model measurement in the *Contextual models* used the complete test data (*n=200*) to obtain the contextual measurement values and perform the training data search. This approach ensured the accuracy measurements reflected the performance of the *Contextual models*, independent of the original measurements to predict the session behaviour.

A Friedman test was run to determine if there were differences in gesture recogniser accuracy between the subject, domain and selection method of the user models. Pairwise comparisons were performed with a Bonferroni correction (p <.00064) for multiple comparisons. Gesture recogniser accuracy was statistically significantly different between the user model conditions, $\chi^2(11) = 32.279, p = .001$. Post hoc analysis revealed statistically significant differences in gesture recognisers accuracy from Generic Shared Contextual (*Mdn*=98%) models to Individual Application Unweighted (*Mdn*=81.1%) (p<.0005) and Group Application Unweighted (*Mdn*=59%) (p<.0005). Figure 8.12 illustrates the gesture recogniser accuracy for each of the model subject, domain and selection method conditions.

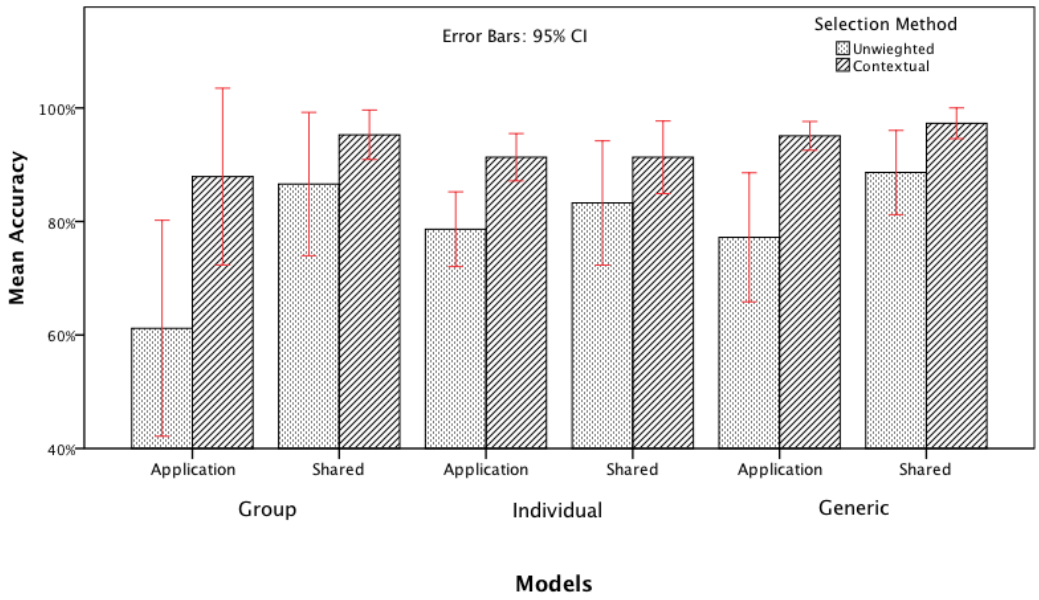

Figure 8.12 Classification accuracy of the gesture recognisers for each of the subject (group, individual, generic); domain (application, shared); and selection method (unweighted, contextual) touch model conditions.

A Kruskal-Wallis test was used to determine whether there were differences in the accuracy of the gesture recognisers between the Default, Unweighted, and Contextual touch model conditions. Pairwise comparisons were performed with a Bonferroni correction (p<.0167) for multiple comparisons. Gesture recogniser accuracy showed a statistically significant difference between the touch models, $\chi^2(2) = 39.78$, p <.001. Post-hoc analysis revealed statistically significant differences in gesture recogniser accuracy between the Contextual (*Mdn* = 96.2%) and Default (*Mdn* = 82%) (p<.001), and Contextual and Unweighted (*Mdn* =85.5%) (p<.001) touch model conditions, but not between the Default and Unweighted (p = .920). These results suggest that the Contextual models have an effect on the performance of the gesture recognisers. Specifically, these results demonstrate that situation-specific user models can improve the touch recognition accuracy of touchscreen

devices for individuals with motor and visual impairments, as illustrated in Figure 8.13.

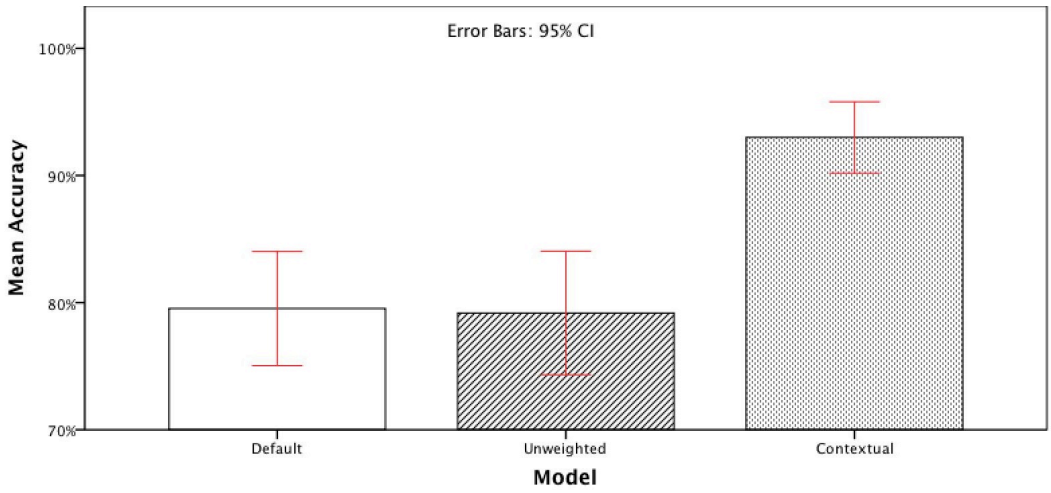

Figure 8.13 Classification accuracy of gesture recognisers for touch model conditions.

8.6.3 Subject of Models

A Kruskal-Wallis test was run to determine whether there were differences in gesture recogniser classification accuracy between the *Subject* conditions. The differences in classification accuracy were not statistically significant within the *Unweighted*, $\chi^2(2) = 1.649, p = .439$ or *Contextual*, $\chi^2(2) = 1.005, p = .605$ models. These results suggest that the subject of the dataset does not affect the accuracy of our touch models, therefore permitting the creation of touch models from the interactions of other users, not specifically from the same stereotypical user group. We have found that when applying the contextual measurements it is actually more beneficial to share data between users, with the results increasing from the individual (*Mdn*=93%) to group (*Mdn*=96.5%) and generic (*Mdn*=98%) conditions. This shows, as more data is made available, that the contextual models are able to locate data that closely mimics the user's current behaviours

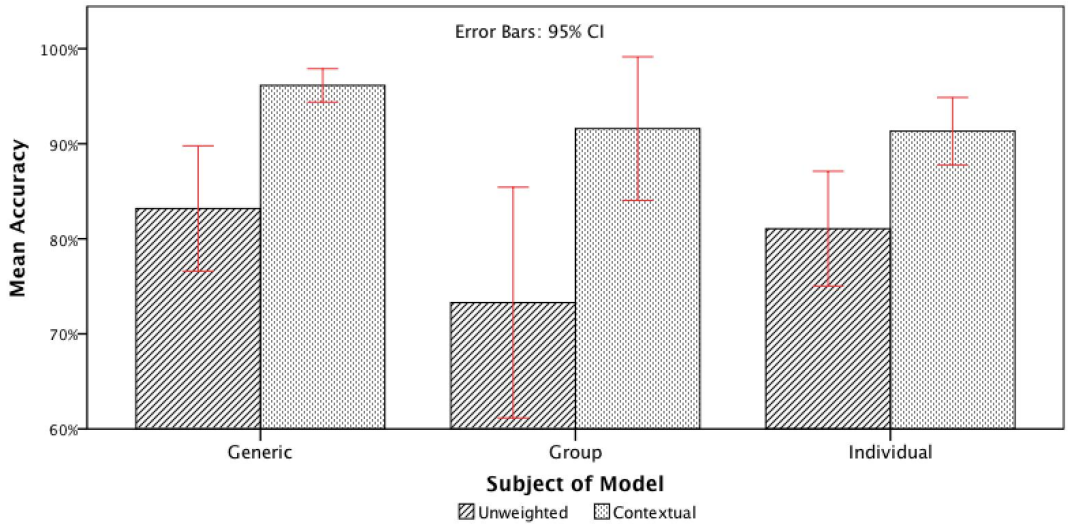

Figure 8.14 Classification accuracy of gesture recognisers for touch models by subject condition

8.6.4 Domain of Models

An Independent-Samples Mann-Whitney U test was used to determine whether there were differences in gesture recogniser classification accuracy between the Application and Shared *Domain* conditions. The classification accuracy was significantly higher for the *Shared* (*Mdn*=92%) than *Application* (*Mdn*=75.1%) or *Unweighted* models, U=719, *z* =3.066, *p*=.002. These results suggest that the domain of the dataset does affect the classification accuracy of our *Unweighted* models. However, related-samples Wilcoxon tests revealed statistically significant differences between the *Default* recogniser and the *Shared, z* =-2.740, *p*=.006, but no significant differences between the *Default* and *Application,* z =-1.607, *p*=.108., therefore supporting the training of models using interaction data *shared* between applications.

No statistical differences in classification accuracy were found within the *Contextual* models between *Shared* (*Mdn*=97.8%) and *Application* (*Mdn=95%*), U=592, *z* =

1.081, p=.280. Related-samples Wilcoxon tests revealed statistically significant differences between the *Default* recogniser and the *Shared,* z =-4.601, p<.001 and *Application,* z =-3.352, p=.001 and *Contextual* models. This suggests that the *domain* condition does not affect the accuracy of our *Contextual* models. This outcome could be the result of the refined selection of training data by the *Contextual* model, minimising the effect of the model domain. Specifically, these results suggest that the contextual models can be trained using either *Application* only, or *Shared* data. Figure 8.15 graphically illustrates the classification accuracy of the domain models.

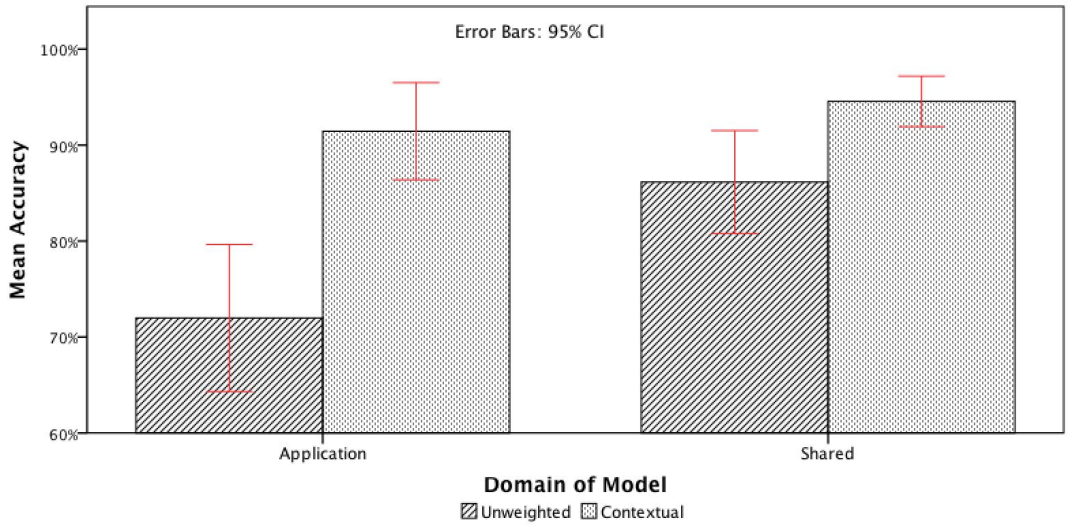

Figure 8.15 Classification accuracy of gesture recognisers for touch models by domain condition

8.6.5 Contextual Measurement Delay

The simulation of the *Contextual model* accuracy was conducted independent of the initial measurements to predict the interaction behaviours and contextual search parameters. However, in a real-world evaluation this method would not have been possible, as the contextual measurements would require time to obtain the appropriate number of gesture instances. Therefore, to compute the effects of the

contextual measurement sample window a delay was applied to the contextual model simulations. The simulation applied the baseline device gesture recogniser to any touches occurring within the delay window. Beyond this point the *Contextual models* were used to classify the interactions. Recogniser accuracy decreased from a delay n=10 (*Mdn* = 93.5%), to n=50 (*Mdn* = 89%) but the differences were not statistically significant, $\chi^2(8) = 8.417$, $p = .394$.

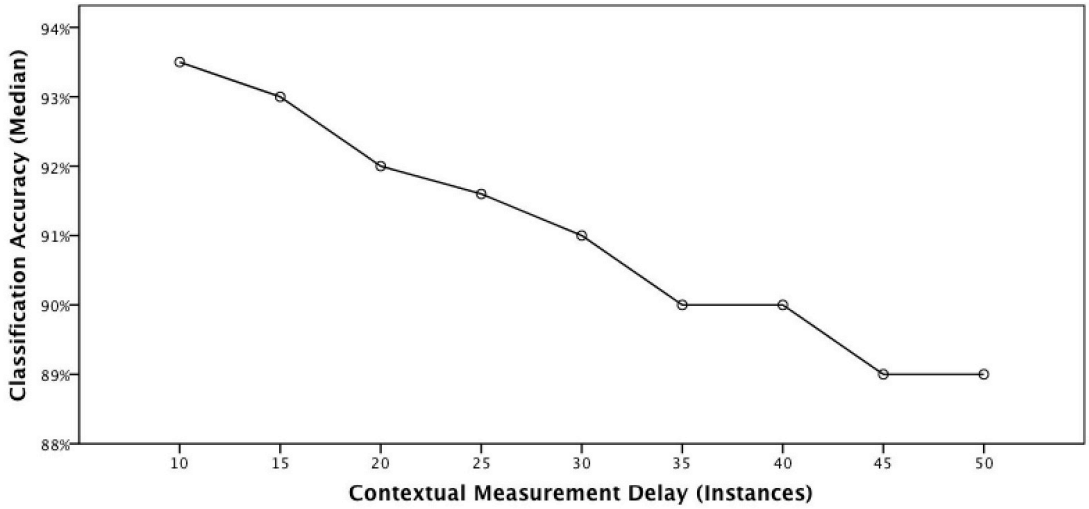

Figure 8.16 Median classification accuracy by contextual measurement delay

8.7 Discussion

Following the simulation and evaluation of the user models the research questions proposed at the beginning of this evaluation can now be answered.

1. *Can shared user models improve touch recognition accuracy over the default gesture recognisers of the devices?*

Results showed touch gesture recognisers using personalised user models could outperform the classification accuracy of the device default gesture recognisers. However, the *Unweighted* models relied on random selection of training data producing highly variable results that would not consistently improve touch recognition accuracy. Using the *Contextual* models to create personalised gesture recognisers resulted in more consistent performance, with significantly better recognition accuracy.

2. *Can we use data from other people (within, and outwith the same stereotypical group) to build shared user models that are more accurate than using an individual's own data?*

Results showed that the *subject* of the training data had no significant effect on the recogniser accuracy. This result was true for both *Unweighted* and *Contextual* models. Moreover, the *Contextual* models achieved higher levels of accuracy with the stereotypical group and generic subject conditions than the individual's own data.

3. *Can we use data from other applications to build shared user models that are more accurate than using an application's own data?*

Results showed that user models trained with the combined data from all three applications performed significantly better than models trained with data from the Sudoku application only within the *Unweighted* selection method. However, no significant effect was measured between *domain* conditions when using the *Contextual* models. The *Shared* user models performed significantly better than the device's default recognisers. This result was true for both *Unweighted* and

Contextual models, suggesting that using data from other applications can improve the accuracy of user models.

4. *Can contextual models improve the accuracy of the shared user models over the Unweighted models?*

The simulation results demonstrated that gesture recognisers trained with *Unweighted* user models, relying on random selection of training data (regardless of *subject,* or *domain* conditions) were not significantly more accurate than the device's *Default* recogniser. However, the *Contextual* models were significantly better than the *Default* and *Unweighted* conditions, supporting the hypothesis that the application of contextual measurements does improve accuracy.

The evaluation computed user models for training size of 50 and sizes 100-1000 in intervals of 100 instances and found no measureable differences between user models trained with greater than 300 instances. However, when Wilcoxon related samples tests were conducted between *Contextual* models and the default recogniser, the models were significantly more accurate using n=50, z =-3.330, p=.001, n=100, z =-4.493, p<.001, and n=200, z =-5.173, p<.001 training instances. Specifically, these results suggest that *Contextual* models could be trained with fewer instances and still provide significant improvements to the baseline gesture recogniser accuracy. Finally, the effects of the contextual measurement delay window were evaluated, revealing a diminishing returns problem with *Contextual modelling.* Larger window sizes are needed to obtain more accurate measurements and

predictions of the session behaviours. However, shorter measurement windows are required to gain the maximum returns of applying the user model.

8.8 Conclusions

The work presented in this chapter investigated the effects of adapting touch gesture recognisers with personalised user models trained from participants' real-world interactions with device applications. In particular, the objectives of this investigation were to evaluate the effects of training user models with data shared between applications, and between users. Moreover, the analysis explored the use of contextual sampling and measurements to source training data harmonious with the present interaction session to create situation-specific user models. Results showed that unconsidered selection of real-world training data, regardless of the originating application or user does not produce significant improvements to the recognition accuracy of touchscreen interactions. However, conducting contextual measurements of the current session to curate the selection of the real-world data resulted in user models and gesture recognisers that performed significantly better than the device's default configurations. Simulations of users' performance with the *Contextual* models produced an average of 10% improvement in accuracy of the gesture recogniser classification. The solution is particularly responsive to the short-term variances within user performance of touchscreen gestures.

Chapter 9. Conclusions

This chapter will conclude the dissertation by presenting the closing discussion of the major contributions and results, the implications and limitations of the approach. Next, the chapter suggests avenues of future research looking to expand on the work within this dissertation. Finally, the chapter presents the researcher's critical reflections on the work.

9.1 Discussion

This research set out to explore the concept of creating user models from real-world data that were responsive to short-term changes, to produce touchscreen interactions that were supportive of individuals' abilities and variances. By leveraging the abstract nature of low-level touchscreen interactions, the research investigated techniques to share and aggregate interactions across applications and users to produce models that provide a holistic representation of user interactions and abilities. Motivated by the need for better touchscreen interactions, this research maintained a realistic approach to investigate solutions based on the functionality within mainstream technologies today, whereby the concepts presented in this dissertation could be implemented tomorrow.

To fulfil the goals of this research, a series of user studies was conducted to examine the characteristics and behaviours when performing touchscreen interactions with mobile devices. The initial exploratory user study combined measurements and observations of touchscreen interactions to identify the characteristics that were common and how they varied amongst users. The findings of the preliminary study,

combined with the previous work within the field, motivated the development of a novel data collection framework, and further laboratory studies to measure and observe the characteristics of individuals with motor and visual impairments. Based on the findings, refinements to the collection framework were made to support the capture of lab quality data from real-world device usage within an in-situ user study. The resulting dataset was used to propose and evaluate novel models that shared data between applications, and users.The simulations demonstrated that user models could leverage contextual measurements of individual situations to improve touch performance.

This section will discuss the major contributions of the research, and the associated implications and limitations of the dissertation.

9.1.1 Contributions and Major results

The objectives of the preliminary user study were to understand the similarities and differences between individuals when naturally interacting with touchscreen mobile devices. Participants were tasked to perform a series of way-finding tasks with the aid of the indoor navigation touchscreen application, enabling the collection and observation of touchscreen interactions free of the restrictive laboratory constraints. Results revealed the diverse range of interaction behaviours between participants, uncovering touch characteristics affecting the success of interactions beyond the task of target acquisition. Moreover, this study exposed the impact of the environment on the success of mobile interactions. Allowing participants to configure interface settings in one space prior to carrying out the navigation tasks in another revealed the importance of short-term changes, and the impact they could have on mobile

interactions. These results motivated the development of the data collection framework to capture user interactions from real-world applications, leading to the proposal of novel gesture recognisers capable of adapting their recognition parameters at use time to meet the abilities and interaction characteristics of the user within the particular context.

Next, laboratory user studies with motor and visually impaired participants were conducted to evaluate the proposed adaptation methods, designed to compensate for variances between user performance of touchscreen interactions and abilities. Similarly, the user studies aimed to understand whether interaction characteristics were shared within stereotypical disability groups. The results demonstrated that the adaptive interfaces significantly improved touchscreen accuracy, and that the models need to be individual as the duration features are user-dependent. Moreover, for some participants the duration features varied between study sessions. Participants reported the impact of medications and fatigue on their performance of similar tasks to touchscreen interaction.

Based on the participant comments and findings of the laboratory evaluations, the research focus was to understand the variable nature of individuals' interaction characteristics. Seeking to capture a continuous representation of a user's abilities, a four-week in-situ user study was proposed to address the limited collection window of the laboratory studies. The objective of the in-situ study was to collect real-world interactions with enough detail to support the laboratory quality analysis of touchscreen performance. Regarding user abilities, the findings demonstrated that interaction behaviours fluctuate between users and between sessions for the same

user. Therefore, it was not enough to build models from an individual's data, user models need to update and respond to the contextual factors and fluctuating abilities of individuals on a per session basis.

In response to the findings of the in-situ evaluation, a novel approach was proposed. This involved leveraging contextual measurements of interaction sessions to refine the data selection used to train the shared user models. Firstly, the method of obtaining the contextual measurements and predicting the users' abilities was evaluated to identify how long to sample user interactions for, and how much training data is required to build accurate models. Next, this chapter explored the effects of training user models from other users' data, and data from other applications. The accuracy of these models was evaluated using simulations from the real-world interaction data collected from the in-situ user study. The shared user models trained using the contextual measurements performed significantly better than the baseline device models and the shared user models without contextual measurements. Furthermore, the simulations demonstrated that there were no significant effects on the accuracy of the gesture recognisers when the contextual shared user models were trained using an individual's data, or using the canonical user data. Therefore, the approach using contextual measurements with shared user models can provide adaptations specific to an individual's abilities and situation.

Although the contextual user models were never evaluated with real users, based on the results from the simulations (Chapter 8) the following process for how SUM adaptations would occur, has been proposed. Figure 9.1 illustrates the proposed interaction workflow of an application using the SUM framework. When the user

opens the application the device default parameters of the gesture recognisers are applied. The SUMClient begins capturing the users interactions, once the framework obtains the required number of gesture interactions (i.e. 10 single taps) it makes a request to the SUMServer for a new user model. The SUMServer uses the received interactions to measure the current interaction behaviours of the user by defining the session features are describe in Chapter 8. These features are then used to perform a search for interaction sessions with contextually similar interaction behaviours. The resulting dataset is used to define the gesture recognisers parameters and returned to the application in the form of a user model. The SUMClient applies the user model to the tap gesture recognisers with no visual interruption or visible change to the user interface of the application. For longer interaction sessions it might be beneficial to repeat this process using the larger dataset of captured interactions for the current session.

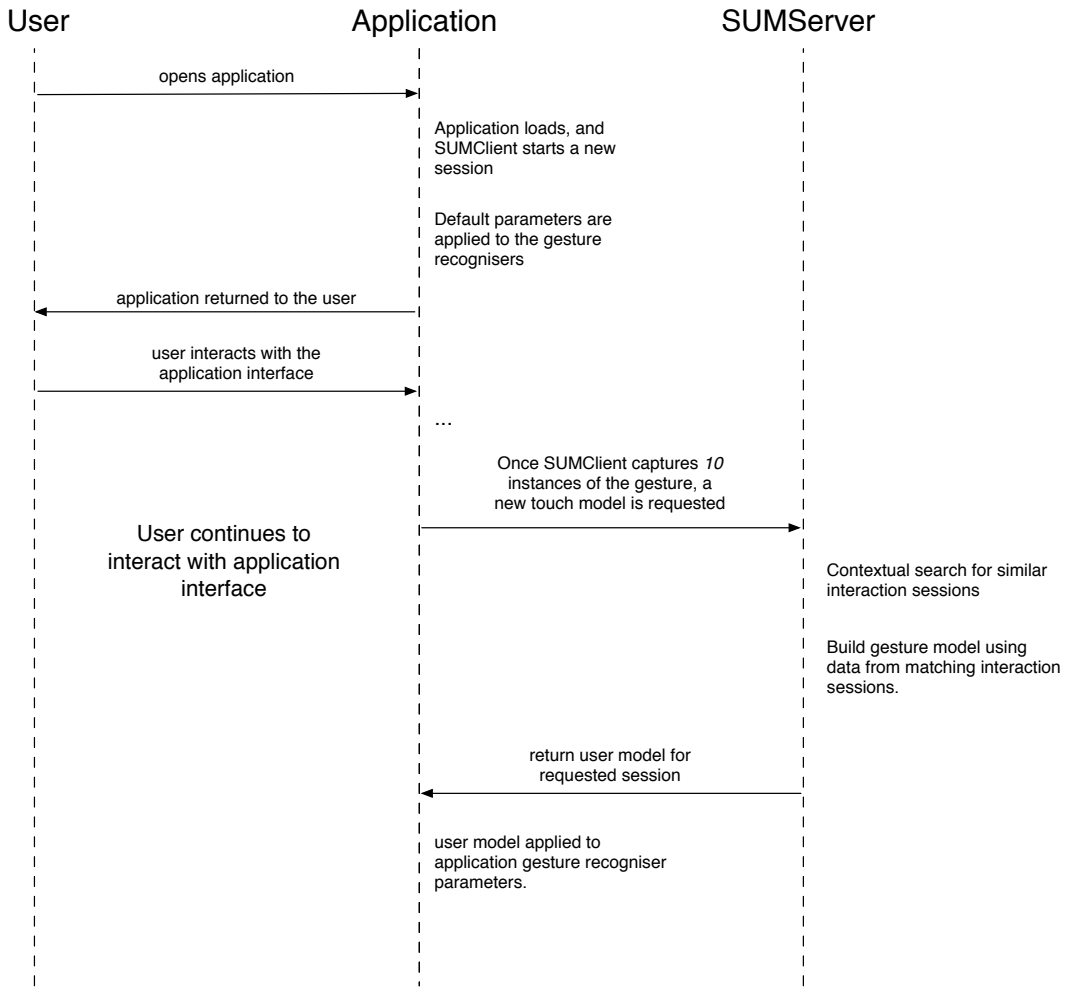

Figure 9.1 Sequence diagram showing the proposed workflow of a user receiving personalised gesture recognisers through the SUM framework.

9.1.2 Benefits

Previous approaches to create user models have relied on the use of semantically meaningless calibration activities to capture measurements of performance, either to select the appropriate stereotypical model to apply, or generate a personalised model for the individual. User modelling with reliance for performance elicitation tasks results in user models naive to the variable nature of individual abilities, thus producing models with a limited shelf life. Furthermore, existing approaches fail to

consider the effects of dynamic environmental factors that do not exist during the measurement process. The work presented within this dissertation has made important steps towards user models representative of the current abilities and situation of an individual. The major benefit in taking this approach is that SUM uses low-level touchscreen interactions from real-world applications to measure user performance, thus removing the need for continuous calibration exercises. Using contextual measurements of the user's current interaction behaviour enables SUM to identify interaction data that closely matches the interaction abilities of the user at that instant; SUM enables accurate user models to be trained using data from other users and applications.

9.1.3 Limitations

Although the approach described within this dissertation was able to accurately model individual users' abilities and situations, it does have some limitations to be explored in future work.

In order to capture the device interactions made by the participants the data collection framework (Chapter 4 and Chapter 6) need to be embedded into each of the experimental applications. Due to the architecture of the mobile device operating systems it was not possible to obtain the low-level sensor interactions and application interface structures on a device level, thus restricting the data collection to the experimental applications only. Therefore, any device interactions participants made with the device outside of these applications were unrecorded. Similarly, should future research projects seek to perform user evaluations of the simulated

touch models (Chapter 8) this implementation method would restrict the effects of the adaptations to interactions made within the experimental applications.

Another related limitation of this approach is the collection of interaction data itself. The method relies on capturing user interactions within real-world applications, such as web browsers; video games; music players and social networking applications. Therefore, any user interactions that occur within these applications would be logged, along with the interface objects being interacted with. This level of detail is required to produce the user models but could similarly be used to gain access to sensitive information about the user. For example, using the keystrokes from the keyboard it would be possible to obtain a user's login credentials, or personal email correspondence. Therefore, the collection of real-world data for user modelling would violate the app store publishing terms and conditions of the mobile operating systems, limiting the mainstream distribution of this approach.

An important limitation regarding the use of real-world data from 'in the wild' installations relates to the measurements of user intent. Chapter 8 discussed the process of leveraging the Sudoku game logic to obtain refined estimations of the user intent behind real-world touch interactions; this same process could not be applied to all real-world applications as they don't all follow a logical interaction pattern. While the refined measurements of intent were not required for the creation of the user models, they were essential for the simulation and evaluation of performance. Providing participants with a mechanism to report unintentional actions as they occur, or methods to refine their input when the system is uncertain of the intent can mitigate this limitation.

Finally, when evaluating the performance of interaction adaptations it is beneficial to perform evaluations with actual users. Evaluations with user studies would have provided an understanding of how the adjustments to the interface were perceived and affected the interaction experience. However, the use of simulations to test the user models provided performance scores for classification accuracy with no insights into the interaction experience implications of situation specific touch models.

9.2 Future Work

User Evaluations of Contextual Models. While the simulated evaluations demonstrated that the contextual models could improve touchscreen recogniser accuracy, it would be beneficial to conduct user studies where the models are being applied in the real-world interactions. This investigation would seek to provide an understanding of how best to adjust user models to ensure consistent user interactions. User feedback would provide useful insights into the acceptance of situation-specific gesture recognisers,;interfaces to control adaptations, and evidence to refine the touch model classifiers.

Contextual Search. The finite size of the dataset for the user model evaluations meant that the contextual search method could effectively compute distances between the current situation of the interaction session and all other existing sessions without large delays. However, if this approach was applied to user studies with larger participant populations or longer collection periods, the number of comparisons required would increase significantly. Therefore, techniques to reduce this search space while maintaining accurate selection of appropriate interaction

sessions would need to be defined in order to provide real-time contextual measurements and searches to support these models.

Beyond Applications. One limitation of the approach was the need to be embedded within the application rather than existing at the operating system level. A direct consequence is the restriction of data collection and interface adaptation to the experimental applications only. Therefore, the device behaviours outside of these applications would default back to the standard gesture recognisers. Moreover, should the user choose not to interact with the experimental applications then no interactions can be collected. Thus, future research should seek to extend this approach and produce methods to capture all interactions made with the device on a system level, while maintaining the detailed access to the interface components that make up the application interfaces. While this was not possible at the time of this work, recent updates to the mobile operating systems suggest it could be viable in the near future. These changes would be crucial to the success of user evaluations with contextual models to ensure the same interaction experience throughout all aspects of device usage.

Refining User Intent. A major challenge of working with real-world data is obtaining accurate measurements of user intent for an action. The evaluations within this dissertation relied on the game logic of the Sudoku application to perform intent classification. However, this method is restricted to applications where user interactions can be modelled and predicted, and thus is limited. Future works should endeavour to support methods of allowing participants to refine or confirm intent for interactions whereby the system is uncertain, rather than always assuming the most

probable action. Another possible method to attain more accurate estimations of user intent would be to model application behaviours, using interaction patterns sourced from large populations. For instance, weighting interface components or gestures based on the common usage amongst all users. This approach would allow a generic set of rules to be defined as opposed to relying on specific classifiers for single applications.

Modelling the Masses. The scope of this research focused on individuals with low levels of vision and motor impairments pertaining to dexterity or unintentional tremor movements. Nevertheless, the approach does not infer or apply any stereotypical context; instead models are constructed based on abilities relating to the interactions. Therefore, it is feasible to suggest that this approach could translate into other domains with alternative sets of abilities. Furthermore, by opening up the approach to wider populations and situations there is the potential to collect a more diverse dataset of interaction eventualities.

9.3 Final Remarks

This dissertation proposed the following thesis:

Sharing data between users and applications can produce models that usefully represent the dynamic needs and abilities of individuals.

Contextual Models produced situation-specific touch models that significantly improved the recognition accuracy of touchscreen interactions. Moreover, models trained from other users' data provided further improvements to the recognition performance. Contextual measurements and weighting of training data has been

shown to produce user models that accurately reflect the current interaction behaviours and abilities of users, therefore, demonstrating that the thesis holds.

Future interaction designers and researchers should aspire to produce solutions that are flexible and accommodating to the individual variances of interaction abilities between both users and situations. Interfaces that are responsive to short-term changes in user performance and interaction behaviour inevitably provide greater accessibility.

References

Allen, R. B. (1990). User models: theory, method, and practice. *International journal of man-machine studies*, *32*(5), 511–543.

Apple. (2009, September 1). iOS Human Interface Guidelines. *developer.apple.com*. Retrieved February 28, 2012, from https://developer.apple.com/library/ios/#documentation/UserExperience/Conceptual/MobileHIG/Introduction/Introduction.html

Brooke, J. (1996). SUS-A quick and dirty usability scale. *Usability evaluation in industry*, *189*, 194.

Buschek, D., Rogers, S., & Murray-Smith, R. (2013). User-specific touch models in a cross-device context (p. 382). Presented at the the 15th international conference, New York, New York, USA: ACM Press. doi:10.1145/2493190.2493206

Cellular-News. (2008, May 8). Touch Screen Market to Reach $3.3 Billion by 2015. *cellular-news.com*. Retrieved April 17, 2013, from http://www.cellular-news.com/story/31048.php

Chapuis, O., Blanch, R., & Beaudouin-Lafon, M. (2007). Fitts' law in the wild: A field study of aimed movements.

comscore. (2012a). *2012 Mobile Future in Focus* (pp. 1–49).

comscore. (2012b, May 11). UK Email Usage Shifting from Computer to Mobile. *comscoredatamine.com*. Retrieved April 17, 2013, from

http://www.comscoredatamine.com/2012/05/uk-email-usage-shifting-from-computer-to-mobile/

comscore. (2013, March 15). Smartphones Reach Majority in all EU5 Countries. *comscoredatamine.com*. Retrieved April 17, 2013, from http://www.comscoredatamine.com/2013/03/smartphones-reach-majority-in-all-eu5-countries/

Findlater, L., & Wobbrock, J. O. (2012). Personalized input: Improving Ten-Finger Touchscreen Typing through Automatic Adaptation

(pp. 815–824). Presented at the CHI '12: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, New York, New York, USA: ACM Press. doi:10.1145/2207676.2208520

Finin, T., & Drager, D. (1986). GUMS: a general user modeling system. Presented at the HLT '86: Proceedings of the workshop on Strategic computing natural language, Association for Computational Linguistics.

Flatla, D. R., & Gutwin, C. (2011). Improving calibration time and accuracy for situation-specific models of color differentiation (pp. 195–202). Presented at the ASSETS '11: The proceedings of the 13th international ACM SIGACCESS conference on Computers and accessibility, New York, New York, USA: ACM Press. doi:10.1145/2049536.2049572

Flatla, D., & Gutwin, C. (2012). SSMRecolor: improving recoloring tools with situation-specific models of color differentiation (pp. 2297–2306). Presented at the

CHI '12: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, New York, New York, USA: doi:10.1145/2207676.2208388

Fleiss, J. L., Levin, B., & Paik, M. C. (2013). *Statistical methods for rates and proportions*. (J. W. Sons, Ed.). John Wiley & Sons.

Froehlich, J., Wobbrock, J. O., & Kane, S. K. (2007). Barrier pointing: using physical edges to assist target acquisition on mobile device touch screens. Presented at the Assets '07: Proceedings of the 9th international ACM SIGACCESS conference on Computers and accessibility, doi:10.1145/1296843.1296849

Gajos, K. Z., Weld, D. S., & Wobbrock, J. O. (2010). Automatically generating personalized user interfaces with Supple. *Artificial Intelligence*, *174*(12-13), 910–950. doi:10.1016/j.artint.2010.05.005

Gajos, K. Z., Wobbrock, J. O., & Weld, D. S. (2007). Automatically generating user interfaces adapted to users' motor and vision capabilities. *Proceedings of the 20th annual ACM symposium on User interface software and technology*, 231–240.

Gajos, K. Z., Wobbrock, J. O., & Weld, D. S. (2008). Improving the performance of motor-impaired users with automatically-generated, ability-based interfaces. *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, 1257–1266.

Gajos, K., & Weld, D. S. (2004). SUPPLE: automatically generating user interfaces. Presented at the IUI '04: Proceedings of the 9th international conference on Intelligent user interfaces, doi:10.1145/964442.964461

Gajos, K., Reinecke, K., & Herrmann, C. (2012). Accurate measurements of pointing performance from in situ observations (pp. 3157–3166). Presented at the CHI '12: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, New York, New York, USA: doi:10.1145/2207676.2208733

Goel, M., Findlater, L., & Wobbrock, J. O. (2012). WalkType: using accelerometer data to accomodate situational impairments in mobile touch screen text entry. *CHI '12: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2687–2696. doi:10.1145/2207676.2208662

Gregor, P., & Newell, A. F. (2001). Designing for dynamic diversity - Making accessible interfaces for older people. Presented at the Proceedings of the 2001 EC/NSF workshop on ….

Guerreiro, T., Nicolau, H., Jorge, J., & Gonçalves, D. (2010a). Assessing mobile touch interfaces for tetraplegics. *Proceedings of the 12th international conference on Human computer interaction with mobile devices and services*, 31–34. doi:10.1145/1878803.1878809

Guerreiro, T., Nicolau, H., Jorge, J., & Gonçalves, D. (2010b). Towards accessible touch interfaces. *Proceedings of the 12th international ACM SIGACCESS conference on Computers and accessibility*, 19–26.

Harper, S. (2007). Is there design-for-all? *Universal Access in the Information Society*, *6*(1), 111–113. doi:10.1007/s10209-007-0071-2

Henze, N., & Boll, S. (2011). It does not Fitts my data! analysing large amounts of mobile touch data. *Human-Computer Interaction–INTERACT 2011*, 564–567. doi:10.1145/1979742.1979604

Henze, N., Rukzio, E., & Boll, S. (2011). 100,000,000 taps: analysis and improvement of touch performance in the large (pp. 133–142). Presented at the Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services, ACM. doi:10.1145/2037373.2037395

Henze, N., Rukzio, E., & Boll, S. (2012). Observational and experimental investigation of typing behaviour using virtual keyboards for mobile devices (pp. 2659–2668). Presented at the CHI '12: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, New York, New York, USA: ACM Press. doi:10.1145/2207676.2208658

Heron, M., Hanson, V. L., & Ricketts, I. W. (2013). Accessibility support for older adults with the ACCESS framework. *International Journal of Human-Computer Interaction*, *29*(11), 702–716. doi:10.1080/10447318.2013.768139

Holz, C., & Baudisch, P. (2010). The generalized perceived input point model and how to double touch accuracy by extracting fingerprints (p. 581). Presented at the CHI '10: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, New York, New York, USA: doi:10.1145/1753326.1753413

Holz, C., & Baudisch, P. (2011). Understanding touch. *CHI '11: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2501–2510. doi:10.1145/1978942.1979308

Hurst, A., Hudson, S. E., & Mankoff, J. (2010). Automatically identifying targets users interact with during real world tasks. *IUI '10: Proceeding of the 14th international conference on Intelligent user interfaces*.

Hurst, A., Hudson, S. E., Mankoff, J., & Trewin, S. (2008a). Automatically detecting pointing performance (pp. 11–19). Presented at the the 13th international conference, New York, New York, USA: ACM Press. doi:10.1145/1378773.1378776

Hurst, A., Mankoff, J., & Hudson, S. E. (2008b). Understanding pointing problems in real world computing environments. Presented at the ASSETS '08: Proceedings of the 10th international ACM SIGACCESS conference on Computers and accessibility.

IDC. (2013, February 21). Mobility Reigns as the Smart Connected Device Market Rises 29.1% in 2012 Driven By Tablet and Smartphone Growth, According to IDC . *idc.com*.

Kane, S. K., Wobbrock, J. O., & Smith, I. E. (2008). Getting off the treadmill: evaluating walking user interfaces for mobile devices in public spaces, 109–118. doi:10.1145/1409240.1409304

Keates, S., & Trewin, S. (2005). Effect of age and Parkinson's disease on cursor positioning using a mouse (pp. 68–75). Presented at the Assets '05: Proceedings of the 7th international ACM SIGACCESS conference on Computers and accessibility, New York, New York, USA: doi:10.1145/1090785.1090800

Koester, H. H., LoPresti, E., & Simpson, R. C. (2005). Toward Goldilocks' pointing device: determining a ‘just right’ gain setting for users with physical impairments (pp. 84–89). Presented at the Assets '05: Proceedings of the 7th international ACM SIGACCESS conference on Computers and accessibility, New York, New York, USA: doi:10.1145/1090785.1090802

Lee, J., & Ha, I. (1999). Sensor fusion and calibration for motion captures using accelerometers (Vol. 3, pp. 1954–1959). Presented at the Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on, IEEE. doi:10.1109/ROBOT.1999.770394

Lee, S., & Zhai, S. (2009). The performance of touch screen soft buttons (pp. 309–318). Presented at the CHI '09: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, New York, New York, USA: doi:10.1145/1518701.1518750

Mace, R. L., Hardie, G. J., & Place, J. P. (1991). *Accessible Environments*.

Macik, M. (2012). Context model for ability-based automatic UI generation (pp. 727–732). Presented at the Cognitive Infocommunications (CogInfoCom), 2012 IEEE 3rd International Conference on, IEEE. doi:10.1109/CogInfoCom.2012.6421947

MacKenzie, I. S. (1992). Fitts' Law as a Research and Design Tool in Human-Computer Interaction. *Human-Computer Interaction*, *7*(1), 91–139. doi:10.1207/s15327051hci0701_3

Montague, K. (2010). Accessible indoor navigation (p. 305). Presented at the ASSETS '10: Proceedings of the 12th international ACM SIGACCESS conference on Computers and accessibility, New York, New York, USA: ACM Press. doi:10.1145/1878803.1878884

Montague, K., Hanson, V. L., & Cobley, A. (2012). Designing for individuals: usable touch-screen interaction through shared user models (p. 151). Presented at the ASSETS '12: Proceedings of the 14th international ACM SIGACCESS conference on Computers and accessibility, New York, New York, USA: doi:10.1145/2384916.2384943

Nicolau, H. (2013). Disabled “R”All: Bridging the Gap between Health and Situational Induced Impairments and Disabilities. *ist.utl.pt*.

Nicolau, H., & Jorge, J. (2012a). Touch typing using thumbs: understanding the effect of mobility and hand posture (pp. 2683–2686). Presented at the CHI '12: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, New York, New York, USA: doi:10.1145/2207676.2208661

Nicolau, H., & Jorge, J. (2012b). Elderly text-entry performance on touchscreens (p. 127). Presented at the ASSETS '12: Proceedings of the 14th international ACM SIGACCESS conference on Computers and accessibility, New York, New York, USA: doi:10.1145/2384916.2384939

Parhi, P., Karlson, A. K., & Bederson, B. B. (2006). Target size study for one-handed thumb use on small touchscreen devices (p. 203). Presented at the the 8th

conference, New York, New York, USA: ACM Press. doi:10.1145/1152215.1152260

Park, Y. S., & Han, S. H. (2010). Touch key design for one-handed thumb interaction with a mobile phone: Effects of touch key size and touch key location. *International journal of industrial ergonomics*, *40*(1), 68–76. doi:10.1145/1409240.1409304

Park, Y. S., Han, S. H., Park, J., & Cho, Y. (2008). Touch key design for target selection on a mobile phone (p. 423). Presented at the MobileHCI '08: Proceedings of the 10th International Conference on Human-Computer Interaction with Mobile Devices and Services, New York, New York, USA: ACM Press. doi:10.1145/1409240.1409304

Pham, C., Plötz, T., & Olivier, P. (2010). A dynamic time warping approach to real-time activity recognition for food preparation. Presented at the AmI'10: Proceedings of the First international joint conference on Ambient intelligence, Springer-Verlag.

Rich, E. (1979). User modeling via stereotypes. *Cognitive science*, *3*(4), 329–354.

Rich, E. (1983). Users are individuals: individualizing user models. *International journal of man-machine studies*, *18*(3), 199–214.

Richards, J. T., & Hanson, V. L. (2004). Web accessibility: a broader view (p. 72). Presented at the WWW '04: Proceedings of the 13th international conference on World Wide Web, New York, New York, USA: ACM. doi:10.1145/988672.988683

Rogers, S., Williamson, J., Stewart, C., & Murray-Smith, R. (2011). Anglepose: robust, precise capacitive touch tracking via 3d orientation estimation. *Proceedings of the 2011 annual conference on Human factors in computing systems*, 2575–2584.

Roto, V., Oulasvirta, A., Haikarainen, T., Kuorelahti, J., Lehmuskallio, H., & Nyyssonen, T. (2004). Examining mobile phone use in the wild with quasi-experimentation. *Helsinky Institute for Information Technology (HIIT), Technical Report*, *1*, 2004.

Sears, A., & Young, M. (2002). Physical disabilities and computing technologies: an analysis of impairments. *The human-computer interaction handbook*.

Sears, A., Lin, M., Jacko, J., & Xiao, Y. (2013). When Computers Fade ... Pervasive Computing and Situationally-Induced Impairments and Disabilities, 1–5.

Trewin, S. (2000). Configuration agents, control and privacy (pp. 9–16). Presented at the Proceedings on the 2000 conference, New York, New York, USA: ACM Press. doi:10.1145/355460.355466

Trewin, S. (2002). An invisible keyguard (p. 143). Presented at the the fifth international ACM conference, New York, New York, USA: ACM Press. doi:10.1145/638249.638275

Trewin, S. (2004). Automating accessibility: the dynamic keyboard (p. 71). Presented at the the ACM SIGACCESS conference, New York, New York, USA: ACM Press. doi:10.1145/1028630.1028644

Trewin, S., & Pain, H. (1997). Dynamic Modelling of Keyboard Skills Supporting Users with Motor Disabilities. In *User Modeling: Proceedings of the 6th International Conference,* A. Jameson, C. Paris and C. Tasso, Eds Springer Wein, New York, (1997), 135–146.

Trewin, S., Keates, S., & Moffatt, K. (2006). Developing steady clicks: (p. 26). Presented at the Assets '06 Proceedings of the 8th international ACM SIGACCESS conference on Computers and accessibility, New York, New York, USA: ACM Press. doi:10.1145/1168987.1168993

Trewin, S., Swart, C., & Pettick, D. (2013). Physical accessibility of touchscreen smartphones. The 15th International ACM SIGACCESS Conference, 19–8. doi:10.1145/2513383.2513446

Vanderheiden, G. C. (2000). Fundamental principles and priority setting for universal usability (pp. 32–37). Presented at the Proceedings on the 2000 conference, New York, New York, USA: ACM Press. doi:10.1145/355460.355469

Vogel, D., & Baudisch, P. (2007). Shift (p. 657). Presented at the CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems, New York, New York, USA: ACM Press. doi:10.1145/1240624.1240727

Wacharamanotham, C., Hurtmanns, J., Mertens, A., Kronenbuerger, M., Schlick, C., & Borchers, J. (2011). Evaluating swabbing: a touchscreen input method for elderly users with tremor (p. 623). Presented at the CHI '11: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, New York, New York, USA: doi:10.1145/1978942.1979031

Weir, D., Rogers, S., Murray-Smith, R., & Löchtefeld, M. (2012). A user-specific machine learning approach for improving touch accuracy on mobile devices (pp. 465–476). Presented at the 25th annual ACM symposium, New York, New York, USA: ACM Press. doi:10.1145/2380116.2380175

Wikipedia. (n.d.). List of iOS devices. (Wikipedia, Ed.)*en.wikipedia.org*. Retrieved April 11, 2013, from http://en.wikipedia.org/wiki/List_of_iOS_devices

Wobbrock, J. O., Kane, S. K., Gajos, K. Z., Harada, S., & Froehlich, J. (2011). Ability-Based Design: Concept, Principles and Examples. *Transactions on Accessible Computing (TACCESS)*, *3*(3), 1–27. doi:10.1145/1952383.1952384

Wobbrock, J. O., Myers, B. A., & Kembel, J. A. (2003). EdgeWrite: a stylus-based text entry method designed for high accuracy and stability of motion. *Proceedings of the 16th annual ACM symposium on User interface software and technology*, 61–70.

Appendices

1. Preliminary User Study - Information Sheet and Consent Form

Accessible Navigation Aids – Information Sheet

FOR QUESTIONS ABOUT THE STUDY, CONTACT: Kyle Montague by email: kylemontague@computing.dundee.ac.uk

DESCRIPTION: You are invited to participate in a study looking at the needs for accessible navigation aids. This will involve taking part in a short exercise either giving navigation directions to and from one location to another within the Queen Mother Building (QMB), or following multimedia directions to navigate from one location to another within the QMB. This will follow with a short questionnaire on your experiences and difficulties throughout the study. The entire study should take no longer than 30minutes.

RISKS: The risks associated with this are minimal. It is not anticipated that you will experience any unusual amount of stress or discomfort as a result of participating in this.

PARTICIPANT'S RIGHTS: If you have read this form and have decided to participate, please understand your participation is voluntary and you have the right to withdraw your consent or discontinue participation at any time without penalty. You have the right to refuse to answer particular questions. Your individual privacy will be maintained in all published and written data resulting from the study. Only key researchers will be able to access all the data collected, including personal details. Other researchers may be able to view anonymous parts of your data.

If you have questions about your rights as a study participant, or are dissatisfied at any time with any aspect of this study, you may contact - anonymously, if you wish – Kyle at kylemontague@computing.dundee.ac.uk

Accessible Navigation Aids – Consent Form

Dear Participant

I would first like to thank you for agreeing to take part in this user exercise.

During the workshop today a camera and Dictaphone may be used for data collection. This is only to allow for notes to be made after the session. All the information that you give us, and the recordings (that is all data), will be stored safely and kept separate from information about your identity. Access to your data is limited to the people involved in this research. If information about you is used in publications or presentations, we will ensure no reference is made to your identity. If a photograph or video-clip is used for presentation, your name will be changed. If you do not wish your likeness to be used in any of the material, your image can be blanked from view.

If you have questions about your rights as a study participant, or are dissatisfied at any time with any aspect of this study, you may contact - anonymously, if you wish – by email: kylemontague@computing.dundee.ac.uk

Thank You
Kyle Montague

I am over 18 years old and have read the foregoing and fully understand the contents thereof. **YES/NO**

I agree to take part in this user exercise and I understand that I can withdraw from this at any time. **YES/NO**

I understand that I am not being judged or assessed **YES/NO**

I understand that I can leave any questions that I do not wish to answer **YES/NO**

I agree for my likeness to be used (i.e. on Video Camera, Audio Recording) **YES/NO**

Print Name _____

Signed _____ **Date** _____

2. Laboratory User Evaluation - Information Sheet and Consent Form

INFORMATION SHEET

We would like to invite you to take part in the SUM research study. Before you decide, we would like you to understand why the research is being carried out and what it would involve for you.

What is the SUM research study?

Our current study is to evaluate the SUM (Shared User Modelling) system for personalising mobile devices to meet individual users access needs. SUM logs interaction data about how you use the mobile touch-screen device. By analysing this data our system is able to make customisations to the software and attempts to improve the usability of the applications running on the mobile device.

Am I being evaluated?

No, we are not testing your abilities to use the technology. The purpose of this study is only to evaluate the effectiveness and accuracy of the customisations made by SUM. So if you encounter any issues or problems please don't hesitate to mention these to the researchers.

Can I choose not to take part?

Yes, it is completely your decision to join the study. If you choose to take part in the study we will then ask you to sign a consent form. We will provide you with copies of the forms to keep.

Can I withdraw from the study?

You are free to withdraw from the study at any time, without giving a reason. Likewise you do not have to answer any questions you do not wish too.

What will I have to do?

You will be asked to take part in two evaluations, which will be carried out within the Queen Mother Building at the University of Dundee. We will provide you with an Apple iPod touch screen device, with a number of applications pre-loaded. These will include the following;

- Target Practice game,

- Alternative television guide

- Indoor Navigation tool

As well as using the iPod touch device we would also like to ask about your experiences, collect your feedback and answer any outstanding questions regarding the study.

Again this is not an evaluation of your abilities, but rather to improve the accuracy and design of the SUM system.

What are the possible disadvantages and risks of taking part?

We see no risks associated with this study and hope that it will be an enjoyable and rewarding experience.

What if there is a problem?

If you have a concern about any aspect of the study, you should ask to speak to one of the researchers, Mr Kyle Montague, who will do their best to answer your questions [phone: 01382 388237 or email: kylemontague@computing.dundee.ac.uk]. You can also speak with the project investigator Professor Vicki Hanson [phone: 01382 386510 or email: vlh@computing.dundee.ac.uk]. IF you are unhappy and wish to formally complain, you can do this by speaking to Professor Janet Hughes, Dean and Head of School

Shared User Modelling – Consent Form

Dear Participant

I would first like to thank you for agreeing to take part in this user exercise.

During the workshop today a camera and Dictaphone may be used for data collection. This is only to allow for notes to be made after the session. All the information that you give us, and the recordings (that is all data), will be stored safely and kept separate from information about your identity. Access to your data is limited to the people involved in this research. If information about you is used in publications or presentations, we will ensure no reference is made to your identity. If a photograph or video-clip is used for presentation, your name will be changed. If you do not wish your likeness to be used in any of the material, your image can be blanked from view.

If you have questions about your rights as a study participant, or are dissatisfied at any time with any aspect of this study, you may contact - anonymously, if you wish – by email: kylemontague@computing.dundee.ac.uk

Thank You
Kyle Montague

I am over 18 years old and have read the foregoing and fully understand the contents thereof. **YES/NO**

I agree to take part in this user exercise and I understand that I can withdraw from this at any time. **YES/NO**

I understand that I am not being judged or assessed **YES/NO**

I understand that I can leave any questions that I do not wish to answer **YES/NO**

I agree for my likeness to be used (i.e. on Video Camera, Audio Recording) **YES/NO**

Print Name _____

Signed _____ **Date** _____

Witness of researcher _____ **Date** _____

3. In-Situ User Evaluation - Information Sheet and Consent Form

INFORMATION SHEET

We would like to invite you to take part in the SUM research study. Before you decide, we would like you to understand why the research is being carried out and what it would involve for you.

What is the SUM research study?

Our current study is to evaluate the SUM (Shared User Modelling) system for personalising mobile devices to meet individual users access needs. SUM logs interaction data about how you use the mobile touch-screen device. By analysing this data our system is able to make customisations to the software and attempts to improve the usability of the applications running on the mobile device.

The more data the system is able to collect the greater prospective customisations that can be made to suit you on an individual basis. This is why the study will run for 6-8 weeks. We hope to collect data during various times of the day and potentially within a number of environments e.g. At home, on the move, in a cafe etc.

Am I being evaluated?

No, we are not testing your abilities to use the technology. The purpose of this study is only to evaluate the effectiveness and accuracy of the customisations made by SUM. So if you encounter any issues or problems please don't hesitate to mention these to the researchers.

Can I choose not to take part?

Yes, it is completely your decision to join the study. If you choose to take part in the study we will then ask you to sign a consent form. We will provide you with copies of the forms to keep.

Can I withdraw from the study?

You are free to withdraw from the study at any time, without giving a reason. Likewise you do not have to answer any questions you do not wish too.

What will I have to do?

You will be asked to take part in a 4-6 week evaluation, which will be carried out remotely. We will provide you with an Apple iPod or Archos tablet touch screen device, with a number of applications pre-loaded. These will include the following;

- Target Practice game,

- Alternative television guide

- Sudoku game

- ToDo app

The device will intermittently prompt you to open the applications and use them for a brief period of time. However if it is not convenient or safe to do so simply reject or ignore this request. Along with these preloaded applications you will be allow to install additional applications, music or video clips and we would encourage you to explore and use any other features of the device.

As well as using the iPod touch and Archos tablet devices we would also like to meet informally throughout the 6-8 week study. A researcher will come out to you either at home or an alternative convenient location to ask about your experiences and address any questions or issues you may have.

Finally after the 6-8 week remote study is complete we would ask you to meet for 1 hour to collect your feedback and answer any outstanding questions regarding the study.

Again this is not an evaluation of your abilities, but rather to improve the accuracy and design of the SUM system.

What are the possible disadvantages and risks of taking part?

We see no risks associated with this study and hope that it will be an enjoyable and rewarding experience.

The only disadvantage of this study would be the time commitment. We understand that 6-8 weeks is a long time, however you would only need to use the device for short periods of time throughout.

What happens at the end of the study?

The analysis of the data will be completed by December 2012. The results will then be published in a PhD thesis as well as academic journals and conferences. If you would like to know the outcome of the study, I will send you a copy of the study report and details of any related publications.

What if there is a problem?

If you have a concern about any aspect of the study, you should ask to speak to one of the researchers, Mr Kyle Montague, who will do their best to answer your questions [phone: 01382 388237 or email: kylemontague@computing.dundee.ac.uk]. You can also speak with the project investigator Professor Vicki Hanson [phone: 01382 386510 or email: vlh@computing.dundee.ac.uk]. IF you are unhappy and wish to formally complain, you can do this by speaking to Professor Janet Hughes, Dean and Head of School [phone: 01382 385195 or email: jhughes@computing.dundee.ac.uk].

What will happen with my information?

All data and information collected during the studies will be stored securely and anonymised. Personal information will only be available to the research team, and will not be kept together with any data, images, audio or video recordings from the study. If your data is used for publications or presentations, no reference to your identity will be made. If any material is suitable for presentation or teaching purposes, we will discuss this with you and ask for your consent.

Contact Details

If you have any other questions relating to your involvement or the the research itself please contact a member of the SiDE research group at the University of Dundee.

Kyle Montague
PhD Student
Email:
kylemontague@computing.dundee.ac.uk
Telephone: 01382 388237

Professor Vicki Hanson
Principle Investigator
Email:
vlh@computing.dundee.ac.uk
Telephone: 01382 386510

Thank You

Thank you for taking the time to read this information sheet and considering taking path within this study.

4. Sudoku Task Sheet (tasks 1 and 2)

TASK 1:

4	1	7	3	6	2	8	9	5
2	9	5	8	1	7	4	3	6
6	3	8	5	9	4	2	1	7
3	5	6	7	2	9	1	4	8
8	4	2	6	3	1	7	5	9
9	7	1	4	5	8	3	6	2
7	6	9	1	8	3	5	2	4
1	2	4	9	7	5	6	8	3
5	8	3	2	4	6	9	7	1

TASK 2

3	8	7	5	9	2	4	6	1
4	6	1	7	3	8	5	2	9
5	2	9	4	1	6	3	7	8
2	9	6	3	5	7	1	8	4
8	7	5	1	6	4	2	9	3
1	3	4	8	2	9	6	5	7
9	4	2	6	7	3	8	1	5
6	5	8	9	4	1	7	3	2
7	1	3	2	8	5	9	4	6

5. Memo Task Sheet

Memo task list
Please add the following items to the memo app, throughout the evaluation period.

Task	Title	Details	Due Date	Reminder	When you did the task Eg. 9:30am 05/11/2012
1	Return books	Take library books back	15:30 Today	15:00 Today	
2	Make a pot of soup	Extra chicken	12:00 Tomorrow	09:30 Tomorrow	
3	Finish report		08:00 Tomorrow	16:30 Today	
4	Feed Goldfish		14:45 Today	14:45 Today	
5	Walk the dogs	Don't forget your raincoat	10:00 Tomorrow		
6	Record X Factor	Remember to vote	19:45 Tomorrow	19:30 Tomorrow	
7	Play a game of Sudoku	Try task 6 or 7	18:00 Today		
8	Send that email	Remember to attach the file	17:00 Today	17:00 Today	
9	Renew car insurance			12:30 Tomorrow	
10	Clean the kitchen		15:30 Today		
11	Make a cake	More chocolate this time	17:30 Tomorrow	17:30 Tomorrow	
12	Water the plants		11:15 Tomorrow	11:15 Tomorrow	
13	Cut the grass			16:00 Today	
14	Haircut Appointment		17:00 Tomorrow	16:30 Tomorrow	
15	Take the bins out		19:00 Today		
16	Paint the fences		17:00 Tomorrow		

Task	Title	Details	Due Date	Reminder	When you did the task Eg. 9:30am 05/11/2012
17	Bake cookies			17:30 Tomorrow	
18	Wash the car				
19	Top up mobile phone		13:00 Tomorrow		
20	Play tennis	Bring an extra racquet	15:30 Today		
21	Clean goldfish bowl		20:00 Tomorrow	18:00 Tomorrow	
22	Check lotto numbers	Fingers crossed	18:00 Tomorrow	18:00 Tomorrow	
23	Water the plants		16:15 Today		
24	Vacuum the house			15:45 Tomorrow	
25	Play a game of Sudoku	Try task 11	20:00 Today		
26	Run a bath		19:00 Today		
27	Get the newspaper		10:00 Tomorrow	09:30 Tomorrow	
28	Watch Jamie Oliver		18:30 Today	18:15 Today	
29	Brush the front path			11:30 Tomorrow	
30	Grit the path	Watch out for the ice	19:45 Today		
31	Make christmas card list		18:00 Tomorrow		
32	Get morning rolls		09:30 Tomorrow		

6. TV Guide Task Sheet

TV Guide Task list
Please fill in the missing information using the TV Guide application. **(Note: some of the shows are on Radio stations)**

Task	Channel	Start time	Show Title	When you did the task Eg. 9:30am 05/11/2012
1	BBC 1	17:00 Today		
2	CH 4	15:30 Today		
3	ITV HD	14:00 Tomorrow		
4	Five	21:00 Tomorrow		
5	Sky Atlantic	14:45 Today		
6	BBC HD	10:00 Tomorrow		
7	Sky One	19:45 Tomorrow		
8	Vh1	18:00 Today		
9	ITV1	17:00 Today		
10	BBC Radio1	09:00 Tomorrow		
11	E4	15:30 Today		
12	Disney	17:30 Tomorrow		
13	Cartoon Network	11:15 Tomorrow		
14	Ch4 HD	22:00 Tomorrow		
15	Sky Sports 1	18:00 Tomorrow		
16	Sky News	14:45 Today		

Task	Channel	Start time	Show Title	When you did the task Eg. 9:30am 05/11/2012
17	Nickelodeon	10:00 Tomorrow		
18	Paramount	19:45 Tomorrow		
19	MTV	18:00 Today		
20	More 4	17:30 Tomorrow		
21	ITV1	21:15 Tomorrow		
22	Dave	16:00 Today		
23	BBC Radio4	16:30 Tomorrow		
24	Five	15:00 Today		
25	BBC2	09:30 Tomorrow		
26	Sky Movies Premiere	16:30 Today		
27	Watch	14:45 Today		
28	Sky One	15:30 Today		
29	Sci Fi	17:30 Tomorrow		
30	BBC6 Music	11:15 Tomorrow		
31	FX	16:30 Tomorrow		
32	Five US	20:00 Tomorrow		
33	Film Four	19:45 Today		