HARDWARE, SOFTWARE AND PEOPLEWARE

UNISA

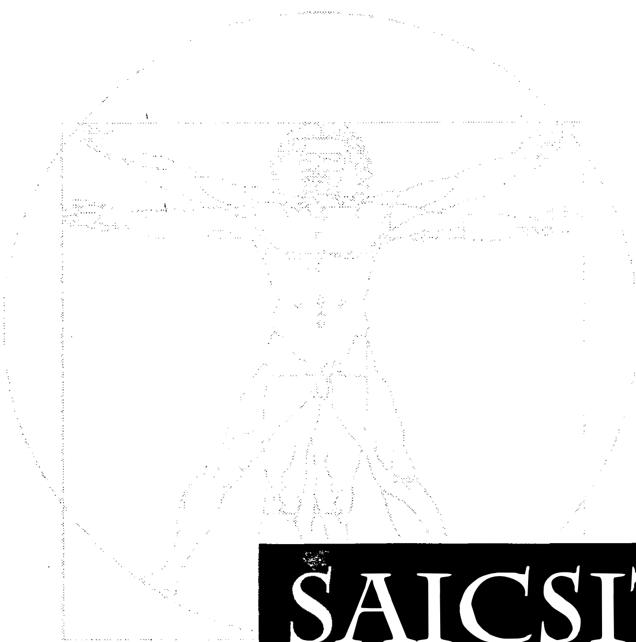
SAICSIT 2001

Edited by
Karen Renaud
Paula Kotzé
Andries Barnard

HARDWARE, SOFTWARE AND PEOPLEWARE

South African Institute of Computer Scientists and Information Technologists Annual Conference

25 – 28 September 2001
Pretoria, South Africa



SAICSIT 2001

Edited by Karen Renaud, Paula Kotzé & Andries Barnard
University of South Africa, Pretoria

Proceedings of the Annual Conference of the South African Institute of Computer Scientists and Information Technologists

First Edition, First Impression
ISBN: 1-86888-195-4

Published by Unisa Press
University of South Africa
P O Box 392, Pretoria, 0003

Cover Design by Tersia Parsons

Editors: Karen Renaud, Paula Kotzé & Andries Barnard

Electronic Publication by the Editors

Printed by Unisa Press
2001

Table of Contents

Abstracts of Electronic Papers

Message from the SAICSIT President

The South African Institute of Computer Scientists and Information Technologists (SAICSIT) was formed in 1982 and focuses on research and development in all fields of computing and information technology in South Africa. Now in the 20th year of its existence, SAICSIT has come of age, and through its flagship series of annual conferences provides a showcase of not only the best research from the Southern-African region, but also of international research, attracting contributions from far afield. SAICSIT does, however, not exist or operate in isolation.

More than 50 years have passed since the first electronic computer appeared in our society. In the intervening years technological development has been exponential. Over the last 20 years there has been a vast growth and pervasiveness of computing and information technology throughout the world. This has led into the expansion and consolidation of research into a diversity of new technologies and applications in diverse cultural environments. During this period huge strides have also been made in the development of computing devices. The processing speed of computers has increased thousand-fold and memory capacity from megabytes to gigabytes in the last decade alone. The Southern African region did not miss out on these developments.

It is hardly possible for such quantitative expansion not to bring a change in quality. Initially computers had been developed mainly for purposes such as automation for the improvement of processing, labour-reduction in production and automation control of machinery, with artificial intelligence, which made great strides in the 1980s, seen as the ultimate field to which computers could be applied. As we moved into the 1990s it was recognized that such an automation route was not the only direction in the improvement of computers. The expansion of processing power has enabled image data to be incorporated into computer systems, mainly for the purpose of improving human utilisation. For most computer technologies of the 1990s, including the Internet and virtual reality, automation was not the ultimate purpose. Humans were increasingly actively involved in the information-processing loop. This involvement has gradually increased as we move into the 21st century. Development of computer technology based not on automation, but on interaction, is now fully established.

The method of interaction has significantly changed as well. The expansion of computer ability means that the same function can be performed far more cheaply and on smaller computers than ever before. The advent of portable and mobile computers and pervasive computing devices is ample evidence of this. The need for users to be at the same location as a computer in order to reap the benefits of software installed on that computer is becoming an obsolete notion. Time and space are no longer constraints. One of the most discussed impacts of computing and information technology is *communication* and the easy accessibility of information. This changes the emphasis for research and development – issues such as cultural, political, and economic differences must, for example, be accommodated in ways that researchers have not previously considered. Our goal should be to enable users to benefit from technological advances, hence matching the skills, needs, and expectations of users of available technologies to their immense possibilities.

The conference theme for the SAICSIT 2001 Conference – *Hardware, Software and Peopleware: The Reality in the Real Millennium* – aims to reflect technological developments in all aspects related to computerised systems or computing devices, and especially reflect the fact that each influences the others.

Not only has SAICSIT come of age in the 21st century, but so has the research and development community in Southern Africa. The outstanding quality of papers submitted to SAICSIT 2001, of which only a small selection is published in this collection, illustrates both the exciting and developing nature of the field in our region. I hope that you will enjoy SAICSIT 2001 and that it will provide opportunities to cultivate and grow the seeds of discussion on innovative and new developments in computing and information technology.

Paula Kotzé
SAICSIT President

Message from the Chairs

Running this conference has been rewarding, exciting and exhausting. The response to the call for papers we sent out in March was overwhelming. We received 64 paper submissions for our main conference and twelve for the postgraduate symposium. We had a panel of internationally recognized reviewers, both local and international. The response from the reviewers was impressive – accepting a variety of papers and *mostly* returning the reviews long before the due date. We were struck, once again, by the sheer magnanimity of academia – as busy as we all are, we still manage to contribute fully to a conference such as SAICSIT.

After an exhaustive review process, where each paper was reviewed by at least three reviewers, the program committee accepted 26 full research papers and 14 electronic papers. Five papers were referred to the postgraduate symposium, since they represented work in progress – not yet ready for presentation to a full conference but which nevertheless represented sound and relevant research. The papers published in this volume therefore represent research of an internationally high standard and we are proud to publish it. Full electronic papers will be available on the conference web site (http://www.cs.unisa.ac.za/saicsit2001/).

Computer Science and Information Systems academics in South Africa labour under difficult circumstances. *The popularity of IT courses stems from the fact that IT qualifications are in high demand in industry, which leads in turn to a shortage of IT academic staff to teach the courses, even when posts are available. The net result is that fewer people teach more courses to more students. IT departments thus rake in ever-increasing amounts of state subsidy for their universities. These profits, euphemistically labelled "contribution to overhead costs", are deployed in various ways: cross-subsidization of non-profitable departments; maintenance of general facilities; salaries for administrative personnel, etc. Sweeteners of generous physical resources for the IT departments may be provided. We have yet to hear of a University in South Africa where significant concessions have been made in terms of industry-related remuneration. At best, small subventions are provided. As a result, shortages of quality staff remain acute in most IT departments – especially at senior teaching levels. What is even worse is that academics in these departments have to motivate the value of their conference contributions and other IT outputs to selection committees, often dominated by sceptical academic power-brokers from the more traditional departments whose continued survival is underwritten by IT's contribution to overhead costs.*[1]

The papers published in this volume are conclusive evidence of the indefatigability and pertinacity of Computer Science and Information Systems academics and technologists in South Africa. We are proud to be part of such a prestigious and innovative group of people.

In conclusion, we would like to thank the conference chair, Prof Paula Kotzé, for her support. We also specially thank Prof Derrick Kourie for his substantial contribution. Finally, to all of you, contributors, presenters, reviewers and organisers – a big thank you – without you this conference could not be successful.

Enjoy the Conference!
Karen Renaud & Andries Barnard

[1] This taken almost verbatim from Professor Derrick Kourie's SACLA 2001 paper titled: "*The Benefits of Bad Teaching*".

Conference Organisation

General Chair
Paula Kotzé

Programme Chairs
Karen Renaud
Andries Barnard

Organising Committee Chairs
Lucas Venter, Alta van der Merwe

Art and Design
Tersia Parsons

Sponsor Liaison
Paula Kotzé, Chris Bornman

Secretarial & Finances
Christa Prinsloo, Elmarie Havenga

Marketing & Public Relations
Klarissa Engelbrecht, Elmarie van Solms, Adriaan Pottas, Mac van der Merwe

Audio Visual
Tobie van Dyk, Andre van der Poll, Mac van der Merwe

Program Committee

Bob Baber – McMaster Univeristy, Canada
Andries Barnard – University of South Africa
Judy Bishop – University of Pretoria
Andy Bytheway – University of the Western Cape
Andre Calitz – University of Port Elizabeth
Elsabe Cloete – University of South Africa
Carina de Villiers – University of Pretoria
Alan Dix – Lancaster University, United Kingdom
Jan Eloff – Rand Afrikaans University
Andries Engelbrecht – University of Pretoria
Chris Johnson – University of Glasgow, United Kingdom
Paul Licker – University of Cape Town
Paula Kotzé – University of South Africa
Derrick Kourie – University of Pretoria
Philip Machanick – University of the Witwatersrand
Gary Marsden – University of Cape Town
Don Petkov – University of Natal in Pietermaritsburg
Karen Renaud – University of South Africa
Ian Sanders – University of the Witwatersrand
Derrick Smith – University of Cape Town
Harold Thimbleby – Middlesex University, United Kingdom
Theda Thomas – Port Elizabeth Technikon
Herna Viktor – University of Pretoria, South Africa
Bruce Watson – Universities of Pretoria and Eindhoven
Janet Wesson – University of Port Elizabeth

Referees

Molla Alemayehu
Trish Alexander
Adi Attar
Bob Baber
Andries Barnard
John Barrow
Judy Bishop
Gordon Blair
Arina Britz
Andy Bytheway
André Calitz
Charmain Cilliers
Elsabe Cloete
Gordon Cooper
Richard Cooper
Annemieke Craig
Thad Crews
Quintin Cutts
Michael Dales
Carina de Villiers
Alan Dix
Dunlop Mark
Elize Ehlers
Jan Eloff
Andries Engelbrecht
Klarissa Engelbrecht
David Forsyth
John Galletly
Vashti Galpin
Wayne Goddard
Alexandré Hardy
Scott Hazelhurst
Johannes Heidema
Tersia Hörne
Chris Johnson
Bob Jolliffe
Paula Kotzé
Derrick Kourie
Les Labuschagne
Paul Licker
Philip Machanick
Anthony Maeder
David Manlove
Gary Marsden
Thomas Meyer
Elsa Naudé
Martin Olivier
Don Petkov
Pekka Pihlajasaari
Nelisha Pillay
Laurette Pretorius
Karen Renaud
Ingrid Rewitzky
Sheila Rock
Markus Roggenbach
Ian Sanders
Justin Schoeman
Martie Schoeman
Elsje Scott
Derek Smith
Elmé Smith
Adrie Stander
Harold Thimbleby
Theda Thomas
Judy Van Biljon
Alta Van der Merwe
André van der Poll
Tobias Van Dyk
Lynette van Zijl
Lucas Venter
Herna Viktor
Bruce Watson
Janet Wesson

Conference
Sponsors

UNISA
ABSA Group Groep
ORACLE
Microsoft
Libra
NRF

Keynote Abstracts

Ubiquitous Computing and Cellular Handset Interfaces – are menus the best way forward?

Gary Marsden [a] **Matt Jones** [b]

[a] *University of Cape Town, South Africa,* gaz@cs.uct.ac.za
[b] *University of Waikato, New Zealand,* always@acm.org

Abstract: *Embedded interactive computer systems, such as those found in cellular handsets, can be hard to use. The combination of small form factor – limited input and output potential – and an increasing feature set, result in devices which confuse novice users. Although most of these devices utilise hierarchical menu structures to mediate the interaction between user and device, we believe that these menus are poorly designed and that other interaction styles may be more appropriate. In this paper we will investigate how well menu design research has been used by current handset manufacturers. We will also propose and report on the success of some new interface designs and finally examine how new Internet technologies, like WML, might be used to further improve the handset's interface.*

Keywords: *User interface design, menus, cellular handsets, WAP, WML*
Computing Review Categories: G.1, H.1.2

1. Introduction

Pervasive ubiquitous computing is becoming a reality. By exploiting UMTS, BlueTooth and other exciting protocols, embedded computers in our fridges can talk to embedded computers in our cars, telling us to stop for milk on the way home. As we evolve towards this new technology, cellular handsets will play a key part in how this technology will evolve – with 413 million cellular handsets sold last year (2000)[1], it is likely that cellular handsets will serve as an introduction to ubiquitous computing for most people. A cursory examination of most current handsets, however, might give us pause before becoming too excited about this new era of information technology.

Ubiquitous computing has been made possible by the continued success in processor and hardware design, which now permits powerful computers to be embedded in devices as small as a cellular handset. Whilst the functional capabilities of these handsets have increased, the way users access their functionality has remained the same – the hierarchical menu is still with us.

Although handset manufacturers have attempted to improve handset menus, as we shall see in the next section, their attempts have been largely cosmetic. If we are to empower the users of ubiquitous computing, then some new form of interaction must be developed.

2. Menus

Menus were originally designed to exploit the fact that humans are better at recognising commands from a list rather than recalling a particular command name from memory. When first introduced, menus provided an easy-to-use alternative to the more prevalent command line systems. Certainly, given the limited keyboard size on cellular handsets, menus represent a significant advantage over any command line system. The constraints in screen size and form factor also favour menu based dialog over a mouse based graphical user interface. Consequently, the reasons for choosing a menu based interaction would seem sound. Therefore all handsets currently support some form of hierarchical menu to access the functionality of the device. All is not well, however.

Techniques, like menus, translated directly from desk-top to hand-held, without fully considering the consequences, can cause interactional problems. The reduced size of embedded computer systems means that interacting with handset menus is more cumbersome than their desk-top counterparts – one study[14] reporting users being up to three times slower when using menus on a small screen. In the case of cellular handsets, this has caused frustration and complaint from many users. Most vocal among these are cellular service providers who are losing revenue as they need to staff support lines. Furthermore, they find it impossible to market vertical services as potential customers cannot configure their handsets to use these premium services.

So what exactly are the problems users of embedded menu systems encounter?

2.1 Potential problems

To be successful, the interface to the functionality of the handset will, like most other systems, need to support both expert and novice users. Considering the expert users first, research has shown that this group of user is able to perform "identity mapping"[2], whereby the user knows the exact name of the option they are searching for in the menu structure. Experts can then quickly scan the screen, until the exact phrase they are searching for appears. This type of searching is very fast and allows experts to rapidly access the function they desire. Furthermore, experts will have learnt the structure of a menu and be able to access a function relatively rapidly in any location [3]. Experts are therefore unlikely to encounter problems in using embedded menu systems. This is not the case for novice users.

Novice users engage in a slower form of searching called "class-inclusion". In this instance, users must make decisions about the higher level menu categories to decide if their target function is contained within a particular sub-menu. For example, users must decide if the function to alter the ringing volume to be found in the "Settings" menu, or the "Tones" menu? Clearly this type of categorisation by the designer (who understands the handset's functionality) can prove problematic to a naive user. When it is not possible for the user to see all the available options (due to reduced screen size) determining the correct class becomes even more difficult – there is extra cognitive load in remembering the previous (currently invisible) classifications.

Assuming the user has navigated to the leaf nodes of the tree, they must perform an "equivalence" search. In this instance the user knows what needs to be done, but does not know the exact phrase used to represent that option. Again, altering the volume of the ring could be described as "Ring Volume", "Volume of Ring", "Tone Amplitude" etc. and requires the user to match their concept with the options presented. Once more, the cognitive load is increased through being forced to recall invisible options rather than compare them directly on the screen.

Another problem for novice users is that of discovering what functionality the device offers. On a handset employing hierarchical menus, this will require the user to perform a complete search of the tree. On a typical handset (say the Nokia 5110, which has 74 functions) this would require the user to make 110 key presses! This figure assumes that the user (a novice) makes no keying or logical mistakes. In our previous experiments[4], we discovered that novice users often pressed the wrong key and could become caught in a sub-menu from which they could not escape.

An interface for novice users must therefore better support comparisons and provide an easier way to discover a handset's functionality.

3. Improving Life

How then might searching be improved for novice users? One response might be to ignore novice users completely. However, the demand for cellular services is still growing and it is safe to assume that there will be many thousands of people each day learning to use a cellular handset for the first time. Not only cellular handsets, but as computing becomes more ubiquitous, we need to develop an interaction technique that will work across a variety of devices. If we are to empower these users, we must find some way to improve the situation for them. We shall investigate a number of ways to improve access for novice handset users.

3.1 Classifications

One way to improve search time would be to improve the categorisations used in the menu classification; perhaps using novices to classify items in a way they feel is appropriate. Although no research specific to cellular handset menus has been conducted, this approach has been attempted in other menu based systems with little success[5]. Even when great care was taken in choosing meaningful classification, users of systems mis-categorised options between 39% and 50% of the time. The evidence from these experiments leads us to believe that it is impossible to produce an ideal classification system for all users.

Another question to ask, then, is how many classifications are appropriate? This question has been asked before in terms of breadth vs. depth trade-offs – is it better to provide a wide range of classifications for comparison at the root, or provide few initial classifications to limit user choice? So far, the majority of research conducted in this area has assumed that the user has access to a full screen and is therefore simultaneously aware of each choice at a given level in the menu structure. With cellular handsets, it is not possible to view options simultaneously, which can have a profound impact on usability.

Research [6] was carried out to consider the impact of reducing the size of the display to a menu system. The smaller the display the fewer options that were presented, with users having to scroll the list to see any options not shown initially. Although users' performance in terms of time to select an option increased as the display size dropped the impact was not dramatic. Real problems occurred, however, when the display was so small that only one option could be displayed at a time – error rates increased dramatically and there was a significant reduction in time taken to access functions. So, handsets which display more than one option at a time (ideally three or more) have similar performance characteristics to desktop systems. A device which only displays one option at a time will be disproportionately more difficult to use.

It would seem that cellular handset designers are unaware of this research as they persist in producing handsets which display only one option at a time. Whilst some handsets are so small that they only support a single line screen (e.g. Ericsson T28) others have a large screen capable of displaying multiple options, yet choose not to do so (any current Nokia or Motorola). To improve interaction, Ericsson has now adopted a menu system which displays three options simultaneously on the screen as seen in Figure 1. *We would advocate that designers of future systems display more than one option at a time.*

Assuming that manufacturers eventually move to displaying more than one option simultaneously[1], we return again to the problem of how to make classifications and providing wide and shallow, or, narrow and deep menu trees. Initial research conducted by Miller[7] and Lee [8] show that wide and shallow trees are more desirable than the narrow and deep variety. More recent research[15] refines this notion to show that concave structures are actually better – i.e. it is important to have a wide choice at the root and at the leaves, but intermediate choices should be restricted. The handsets we investigated do indeed follow this concave structure.

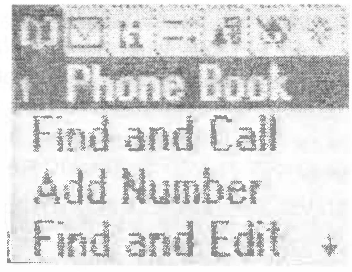


Figure1 – Display from Ericsson R320 showing menu title and 3 options

3.3 Reducing key presses

We have already noted how many key presses are required to access every option in a cellular handset menu. Staying with the Nokia 5110 as our example, we can calculate that the average number of key presses to access a function is 8.2, with a maximum of 14. To improve this count, Nokia introduced wrapped menus (see Figure 2), so than when the user moves beyond the end of the menu, they are automatically shown the first option in the menu. In user experiments we have conducted [4], this feature caused problems with novice users, who would become stuck in a lengthy menu and loop through the options until they gave up in despair. Further research is required to determine if this problem could be eliminated by displaying more than one option at a time, or reducing the maximum number of options to seven (in an attempt to exploit human short term memory [7]). Certainly, looping menus as they exist currently, cause huge problems for novice users.

3.4 Visualisation

The benefits of visualisation of state in interfaces is well understood. Therefore, one way to improve usage of menus, and help avoid the type of problem found with looping menus, is to give the user visual feedback about where in the menu structure they are. There are several ways in which this may be attempted:

3.4.1 Icons

In the devices examined as part of this work, icons were found to exist in two formats:

[1] This seems probable as most manufacturers are producing prototypes with larger touch screens.

- Isolated icons
- Context icons

Isolated icons are those used to augment understanding of a particular menu items. For instance, the Nokia menu systems, since the 5110 model, have displayed an icon beside each of the root level menu options (see Figure 3). It is not at all obvious what purpose these icons serve, as they are not used in any other context and cannot be manipulated in the same way as icons in a WIMP environment. More recent releases of Nokia handsets include animated versions of these icons. Research conducted on animated icons for desktop systems suggest that they are most useful to explain some action or verb [9]. However, of the root level options which have animated icons, only one option is a verb – Call Divert. Even with this option, the animation adds little to understanding the role of the menu as it shows an arrow ricocheting off a small picture of the handset. From our analysis, we can only conclude that isolated icons serve as a marketing feature and add little (if anything) to the usability of the handset.

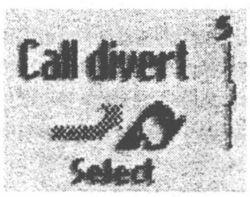


Figure 3 – Image of Nokia root level icon

Context icons are used to highlight a particular choice from a set of alternatives. Rather than showing a single menu option per screen, context icons can be used to display the full set of alternative choices on a single line – the compact icons can be fitted on the screen where the larger text representations cannot (see Figure 4). This type of icon has been used in a curious way in the current range of Ericsson handsets. Rather than exploit these icons to reduce the amount of screen real-estate required, the icons are used in conjunction with the text description of each menu option. Whilst redundant information is helpful to users, the screen space could, perhaps, have been used in more helpful ways: to provide an extra menu option or a scrolling help line, for example. When a sub-option is selected, the icon disappears, meaning that the longer text name is used at the top of the screen to describe the sub-menu. Retaining the icon would be of particular use for providing context in sub-sub menus.

3.4.2 Context information

For novices using a menu, it is essential that they are provided with some form of feedback about where they are within the structure in order to navigate successfully. The limited screen resources of the cellular handset make this a much more difficult task than with desktop based menu systems. Given that some of the handsets we examined nested menus up to four levels deep, the problem of navigation becomes all the more complicated.

In the handsets we examined, Nokia provided the most information about location in a menu structure – not only depth choices, but feedback on the current level. The least information was provided by the Ericsson handsets, which only showed the most recent category choice. This is curious as Ericsson go to extra lengths to provide smooth scrolling when changing menu levels to provide users with as much contextual and spatial information as possible.

One vital piece of information which is missing from these visualisations is feedback about which options in the menus are branch nodes (the selection of which will display another menu) or leaf nodes (the selection of which will access a function). From desktop menus we already have an ellipses (or triangle) convention to denote the difference – leaf nodes have no ellipses beside the name. This type of information is important to novice users exploring a menu structure – they will be more likely to explore the structure if they know their exploration will not affect the handset.

3.4.3 Manuals

Figure 4 – Icons showing context. From left to right: Phone book, Messages, Information, Settings, Extras, WAP, Shortcuts. Phone book is currently selected. (Taken from an Ericsson 320)

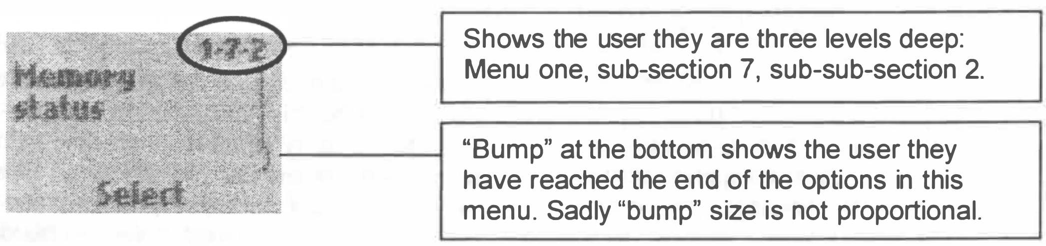


Figure 5 – Feedback from Nokia handset on user location within the menu structure

Manuals for cellular handsets are really of limited usage due to the fact that the manual is usually larger than the device itself. As the point of cellular communication is mobility, it is unlikely that users will carry the manual with the device. Furthermore, research by Youngs[10] shows us that for younger users (under 35), they are *less* likely to complete a task if they use the manual.

On-line manuals, however, can be much more successful. Here, if a user scrolls to a menu option and does not select it, a scrolling description of that option appears on the screen. For example, Lee *et al* [5] found that adding extra information to menu options could reduce errors by up to 82%. On-line help was applied in a seemingly random fashion for the handsets we examined – help was provided on a per-model basis and was not consistent to a particular manufacturer.

3.4.4 Summary

From our investigation we have seen that there are problems in using hierarchical menus to support users of embedded interactive systems. Hierarchical menus rely on the user understanding the designer's classification of functions, something that is certain to cause problems. Menus are weak in supporting novices' exploration of a handset's features – they require too many key presses. Furthermore, the poor visualisation does not always support exploration of the menu. Manufacturers also seem to concentrate their developments on the aesthetic of the menus, rather than improving usability. It is time that an alternative design was found!

4. Alternative designs

A data structure which requires an average of 8.2 key presses to access a given function seems somewhat sub-optimal. Treating this as a computing science problem, one way to improve the menu tree is to re-structure it as a balanced binary tree. Users searching for a menu item would navigate on the alphabetic order of the function name they were searching for. At each node in the tree, users would either select the function name at that node, or choose to navigate down the node's left or right branch. By classifying functions by name we are removing the problems of assessing class inclusion, but other problems remain.

For a given handset (again, as an example we use the Nokia 5110 with 74 functions) a binary tree solution would reduce the average cost of selection from 8.2 key presses to 5.4 key presses. Furthermore, the worst case search path is reduced from 14 presses to 7 presses. However, by maintaining the hierarchical tree structure, the task of visiting every node in the structure is still daunting, requiring the user to make 148 key presses. This scheme also has the problem of only supporting identity searching, further increasing the problems for novices.

Of course, to remove the navigational difficulties of a tree structure, we could flatten the structure to a linear list. This would allow users to visit every function with only 74 key presses; what is more, the key being pressed would be the same every time. However, with a list, the average search time is 37.5 key presses and the worst case search requires 74 key presses. So whilst the list can support exploratory behaviour, it is poor at directed searching.

The best solution would seem to lie in a synthesis of the two approaches.

One solution we developed, exploited the fact that most mobile telephone keypads have up to three alphabetic letters associated with each key. So, on key '1' you can find the letters 'A B C','2' has 'D E F' and so on. (Some phones vary in their key allocation, but this is not relevant to our approach.) Users simply had to 'spell-out' the function they wished to access by pressing the appropriate numeric keys.

Normally, when words are spelt out, the user will press key 1 once to get A, twice to get B or three times to get C. In our approach, the key is only ever pressed once, and it is allowed to mean A or B or C. Thus there is some initial ambiguity as the user starts to press keys to spell a function name. With each new key press, using a standard computer science technique known as hashing, the system displayed the best set of function name matches.

For example, if the user wished to access 'Call Divert' they would begin by pressing '1' followed by '1' (meaning 'C', 'A' the first two letters of the function name) and the system would display a scrollable list containing choices such as 'Call Divert', 'Call Identification' and 'Call Barring.' If any other combination of the keys — in this case ABC followed by ABC — also started function names ('Battery Condition'?), they would also be displayed. As soon as the required function appeared in the best match list it could be selected directly by the user without any further input: the user did not have to spell out the entire function name! To overcome the equivalence search problem, we allowed the words of the function name to be entered in any order: so, for example, Call Divert and Divert Call were both permitted, and the user would probably prefer Divert, since it is unambiguous.

Besides using the numeric keypad to access the functions, users could also scroll the list using the scroll keys. Providing access in this way supported exploratory behaviour as efficiently as possible. In effect, this final solution is similar to a B+Tree. This type of tree supports both sequential searching of leaf nodes (in our case this is provided using the scroll keys) and direct searching via an index (in our case, the hashing supported by the numeric key pad).

Analysis of our solution showed that average search time was reduced from 8.9 to 3.1 key presses. This theoretical result was also backed up by user experiments, which showed that there was a statistically significant reduction in key presses between users of our system compared to users of a standard handset[4] (mean reduction of 5 key presses).

So, by restructuring the menus in more fundamental ways than in current commercial handsets, we can decrease the number of key presses required to access a function and facilitate exploration for novice users. However, these solutions ignore the wider question of "Do we need a user interface at all?"

It has been argued that the whole notion of a "User Interface" is fundamentally flawed – the user should enjoy seamless interaction without being aware of any intermediate layer. This is, of course, not always possible and is particularly exacerbated for embedded computer systems which are constrained by reduced form factor. However, recent developments in mobile computing allow us to investigate another possibility – replacing the menu system with WML Web pages.

5. Graphs, not hierarchies

Once again we re-visit the fundamental problem with hierarchical menus – classification. The example we gave earlier was for the location in a menu system of the function to alter ringing volume: should it be in "Phone Settings" or "Tones"?[2] What if we placed that option in both locations? By doing that, we start to move away from the 1-to-many relationship of hierarchical menus to a many-to-many graph. So are graphs better than menus?

In his paper[11], Alexander argues that humans cannot work with imposed hierarchies. Certainly, the error rates from the menu classification experiments[5] would confirm this argument. Furthermore, it is the attempt to break free from this type of hierarchical thinking which motivated Tim Berners-Lee to develop the World Wide Web. He attributes[18] the success of the WWW to its ability to allow information to be joined according to a user's perception. Furthermore, the simplicity of HTML allowed users to restructure any collection of information as they saw fit.

In fact, the success of the WWW has meant that, in Windows at least, the traditional desktop metaphor is being eroded to be replaced by a browser. Other research[16] has shown how the interface for desktop computers can be completely replaced with browser technology. If the browser works for the desktop computer, then what about embedded computers?

[2] In case you are interested, for the Nokia 5110 the choice was "Tones"

5.1 WAP-menus

When introduced, WAP was criticised for being cumbersome and hard to use[12]. Certainly, when compared to desktop Web surfing, this is certainly the case. More recent work[17] into the usability of WAP and WML, however, is shown that a lot of the initial criticism was based on a poor understanding of the nature of mobile Web access. If the application is tailored properly, then WAP can be an effective way of accessing information. Statistics from the UK [13] show that by July 2000 7% of the population were accessing the Internet via WAP, compared to just 1% one year previously. Given the effort that has already gone in to supporting Internet access on cellular handsets, it is likely that all future handsets will support some level of browser.

If handsets do have a browser installed, then it would seem sensible to do away with the menu structure and replace it with a series of WML decks. By doing this, we free the user from having to learn two types of interface – the navigation techniques they learn for the browser can be transferred to navigating the functionality of the handset. (If successful, this would overcome the problems discovered by Heyler[19], who noted the confusion users experienced when required to change navigation techniques between menus and WAP sites) . To investigate the possibility of providing a WAP interface, we have built a number of prototype systems.

5.2 WML prototypes

The simplest way to replace menu systems is to create WML pages which correspond directly to existing structures. We have already built such a system based on the Nokia 5110, as shown in Figure 6. This prototype presents the user with a home page providing access to local information, or a remote site. If the user selects "local information" they are presented with the WML pages replacing the menu system. In this way, the menu becomes just another site accessible through the browser. The only interactional benefit of this prototype, however, is that the navigation keys and paradigm for the menu system are identical to those required for a WML browser.

To improve the interaction further, we modified the WML to present as many options as possible on the screen at any one time. We then used indentation to provide the user with context information about their choices (see Figure 7). In this way, we have created a system which exploits the work put in to improving hierarchical menus and keeps the navigational benefits of the previous prototype.

Both of the prototypes described above are based on the structure of current menuing systems. Re-using the structure in this way allows current handset users to transfer their knowledge to the browser based system. However, as the options are presented as WML pages, it would be straightforward for handset manufacturers to provide users with a WML editor to restructure the menu system any way they choose. This would allow users to exploit the benefits of a graph based structure as discussed in the previous section.

6. Conclusions

Within this paper, we have questioned the approach of using hierarchical menus to access the functionality of embedded computer systems. Whilst menus may have been appropriate when this functionality was limited, the increasing power of microprocessors means that the functionality of devices has been increasing steadily, but little work has been done to ensure the interface has kept pace.

Starting with research into hierarchical menus, we saw that there was usability research which manufacturers could use to improve the interaction between novice users and hierarchical menus. However, the application of this research is limited and real improvements could only be gained through abandoning the menu structure. Alternative structures were presented and finally a structure based loosely on a B+Tree was proposed which had significant advantages when conducting directed searches and exploratory searches.

Finally, we investigated the use of WAP as a way of removing the interface altogether. Once users have learnt to navigate with the WML browser, they can use it to either modify their handset settings or to access WML sites on the Internet. Whilst this solution does not offer the advantages in reduced key presses that the B+Tree does, it provides a more familiar interface and lends itself to alteration by the user. As handsets increase in functionality and mobile internet access becomes more common, we believe that a WML based interface will be the best way to support users of ubiquitous computing devices.

7. Acknowledgements

The authors would like to thank the honours class of 2001 at UCT for helping with the analysis of handsets and Lindikhaya Ntshinga for developing the WML prototypes.

8. References

[1] World cellular stats from cellular.co.za: http://www.cellular.co.za/4q2000_handset_market_shares.htm

[2] Paap, K.R. Design of Menus Handbook of Human-Computer Interaction. Chapter 10, pp. 205-235. 1988

[3] Card, S.K. User Perceptual Mechanism in the Search of Computer Command Menus. Proceedings of Human Factors in Computer Systems. pp. 190-196. 1982

[4] Marsden, G., Thimbleby, H., Jones, M. & Gillary, P. Successful User Interface Design from Efficient Computer Algorithms. Proc. ACM CHI2000 Extended Abstracts pp. 181–182. 2000

[5] Lee, E., Whalen, T., McEwen, S. & Lantremouille, S. Optimising the Design of Menu Pages for Information Retrieval. Ergonomics, 77 pp. 1051-1069. 1984

[6] Swierenga, S. J. Menuing and scrolling as alternative information access techniques for computer systems: interfacing with the user. Proc. Human Factors Society 34th Annual Meeting, pp. 356–359. 1990

[7] Miller, D. P. The Depth/Breadth Tradeoff in Hierarchical Computer Menus. Proceedings of the Human Factors Society 25th Annual Meeting pp. 296-300. 1981

[8] Lee, E. & MacGregor, J. Minimising User Search Time in Menu Retrieval Systems. Human Factors, 27 (2) pp. 157-163. 1985

[9] Baecker, R., Small, I. & Mander, R. Bringing Icons to Life – Use of Familiar Things in the Design of Interfaces. Proceedings of ACM CHI'91 Conference on Human Factors in Computing Systems pp. 1-6. 1991

[10] Youngs, E. Evaluating the Impact of Application, Ergonomic and Process Design on Handset Success. Proceedings of User Interface Design for Mobile Terminals, Section 1. 1998

[11] Alexander, C. A city is not a tree. DESIGN, 206, pp. 46-55 1965

[12] Nielsen, J WAP backlash (2000) Alertbox 09/07/2000 at www.useit.com/alertbox/000907

[13] UK National Statistics Organisation, quoted in article on BBC Web site: http://news.bbc.co.uk/hi/english/business/newsid_1245000/1245793.stm

[14] Han, S.H. & Kwahk, J. Design of a Menu for Small Displays Presenting a Single Item at a Time. Proceedings of the Human Factors and Ergonomics Society 38th Annual Meeting v.1 pp.360-364 1994

[15] Norman, K. L., & Chin, J. P. The effect of tree structure on search in a hierarchical menu selection system. Behaviour and Information Technology, 7, pp. 51-65. 1988

[16] Rice, J., Farquhar, A., Piernot, P. & Gruber, T. Using the Web Instead of a Window System. Proceedings of ACM CHI 96 Conference on Human Factors in Computing Systems pp.103-110. 1996

[17] Jones, M., Buchanan, G., Marsden, G. & Pazzani, M. Improving Mobile Internet Usability. Proceedings WWW'10, Hong Kong. 2001

[18] Berners-Lee, T. Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web. Harper Business. 2000

[19] Heylar, V. Usability Issues and User Perspectives of a 1st Generation WAP Service. Proceedings of the Wireless-World Symposium, Surrey University, UK. 2000

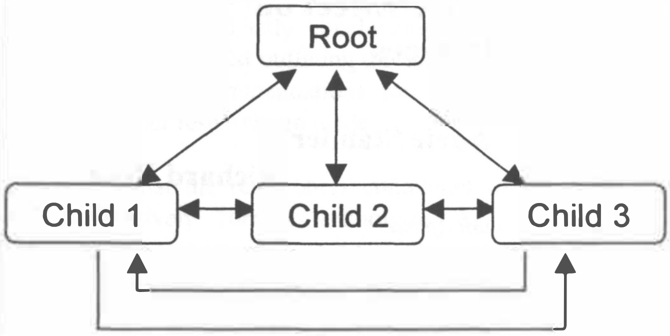


Figure 2
Interface menu hierarchy. Scrolling right from the third child returns to the first child. Scrolling left from the first child moves to the third child.

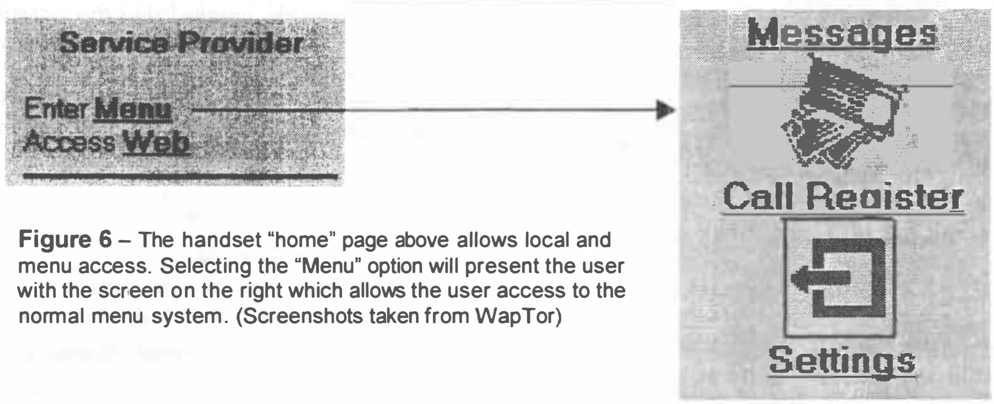


Figure 6 – The handset "home" page above allows local and menu access. Selecting the "Menu" option will present the user with the screen on the right which allows the user access to the normal menu system. (Screenshots taken from WapTor)

Figure 7 – Menu allows several options to be seen simultaneously. Also context information can be used for navigation as in HTML based Web site