

User Interface for a PCS Smart Phone

Shankar Narayanaswamy
Bell Laboratories
791 Holmdel-Keyport Road
Holmdel, NJ 07733 USA
shankar@bell-labs.com

Jiaying Hu
Bell Laboratories
600 Mountain Avenue
Murray Hill, NJ 07974 USA
jianhu@bell-labs.com

Ramanujan Kashi
Bell Laboratories
600 Mountain Avenue
Murray Hill, NJ 07974 USA
ramanuja@bell-labs.com

Abstract

New multimedia smart phones for wireless voice and data communications are now available. In this paper, a new user interface for a Personal Communications Services (PCS) smart phone is described. The user interface is designed around the unique characteristics of mobile wireless applications. Both the physical characteristics of the phone and the way users interact with applications are unique. The phone has two LCD touch screens in a clamshell casing. It features pen input, handwriting recognition, and a graphical user interface supported by the Inferno operating system. A dedicated writing area and automatic text focus flow make form-based applications particularly easy. The smart phone incorporates signature verification for biometric authentication. Apart from being a full-featured cellular voice phone, the device also supports a number of applications including web browsing, electronic mail, fax communications and a directory application which allows users to retrieve contact information.

1. Introduction

Cellular telephony has grown by leaps and bounds. In the last decade, growth rates have been between 35% and 60% per year in the United States[2]. During this time prices for cellular service have dropped [16]. There are also a growing number of Cellular Digital Packet Data (CDPD) users. Recent digital cellular telephone systems feature data channels communicating at up to 14.4 kbits/sec [3] [7] [8].

The Internet has experienced tremendous growth in the past few years. Internet applications include e-mail, network news and Web browsing. On top of that, internet commerce is experiencing explosive growth.

Several Personal Digital Assistants (PDAs) use handwriting recognition in the user interface. This technology has become very effective due to advanced algorithms that attain high recognition accuracies for all users. Several vendors supply handwriting recognition

engines, some of which use a modified alphabet that the user must learn.

Given the developments above, it is desirable and feasible to build a multi-function mobile wireless terminal and the infrastructure to support applications that require access to remote data. The terminal functions primarily as a cellular voice telephone but can store data, run PDA applications and browse the Web.

This paper describes the design of a user interface for such a smart phone. It addresses both hardware and software design, which are necessarily interdependent if we hope to achieve a good user experience. The smart phone builds upon the success of cellular telephones by adding data functionality to a cellular voice telephone. The primary applications support voice and data communications, and the user interface is designed around them.

2. Previous work

The Nokia 9000 Communicator [11] is a full-featured GSM telephone. It is capable of sending faxes, e-mail, and short messages. Internet access and personal organizer functions are supported. The 9000 has a clamshell design. When closed, it operates as a normal cellular telephone. When open, it reveals a LCD display on one side and a keyboard on the other.

There are three aspects of the 9000's design that are relevant to the following discussion. The keyboard is very small so users have a hard time fitting their fingers on it. Keyboard buttons are used for on-screen navigation, which is not as efficient as a pointing device. The screen is small so it displays only a small amount of information at a time.

The AT&T PocketNet telephone [1] provides cellular access to the Internet. It has a 5-line by 20-character display. The aspects of this device that are relevant to the following discussion are its small screen area and the lack of an input mechanism. To display text and graphical data effectively, a larger bit-mapped screen is required.

There are many PDAs on the market [13], some with keyboards and others with pen input. All have small screens. Most support an external modem for wired data communications. The most popular PDA today is the 3Com PalmPilot, which holds 66% of the handheld computer market [15].

The Berkeley InfoPad project [10] demonstrated a high-bandwidth wireless multimedia terminal. It uses handwriting and speech recognition in the user interface and runs the recognition algorithms on fast servers on the wired network. This is possible because the wireless connection is fast enough to send high-quality pen and audio data to the recognizers with acceptable delay.

3. Applications

The smart phone is designed mainly for communications applications. The “minimum package” for the phone to be useful is directory, electronic mail and Web browsing. Secondary applications in information retrieval and PIM are also supported.

The directory allows users to retrieve telephone numbers, URLs, e-mail addresses and other contact information from the white pages, yellow pages, corporate internal directory, or a personal directory. Name search can be used to query the directory; the contact so obtained can then be used directly by the telephone or the Web browser.

Access to electronic mail (e-mail) is one of the biggest reasons that laptops are taken on business trips. E-mail capability is necessary in any mobile personal communications terminal and is a primary application.

Web browsing is the single largest user of bandwidth on the Internet today. Not only is the Web browser useful for retrieving and viewing remote documents, it is also an excellent interface for document searching and viewing, including incoming faxes.

The smart phone supports most other applications but the user interface design was not optimized for them. The secondary applications have some requirements that conflict with the primary applications. For example, an electronic notepad would benefit from a cursive handwriting recognizer but the primary applications require printed handwriting recognition. We believe that it is better to support a few integrated applications very well than many disparate applications in a mediocre manner.

4. Smart phone hardware design

The smart phone is housed in a clamshell casing. When closed (voice mode), it looks and feels similar to a modern cellular telephone (see Figure 1). When open (data mode)

it reveals two touch screens hinged at the edges to achieve a larger active screen area for the same small inactive footprint. The hinge is made horizontal and as narrow as possible to minimize disruption of the visual flow of the logical screen from one physical screen to the other. The screens operate in landscape mode to reduce or eliminate horizontal scrolling. Each physical screen is as large as possible within the overall size of a cellular telephone: a 6” x 2.5” phone supports 640 x 240 screens at 0.2 mm dot pitch (5” x 1.9”).

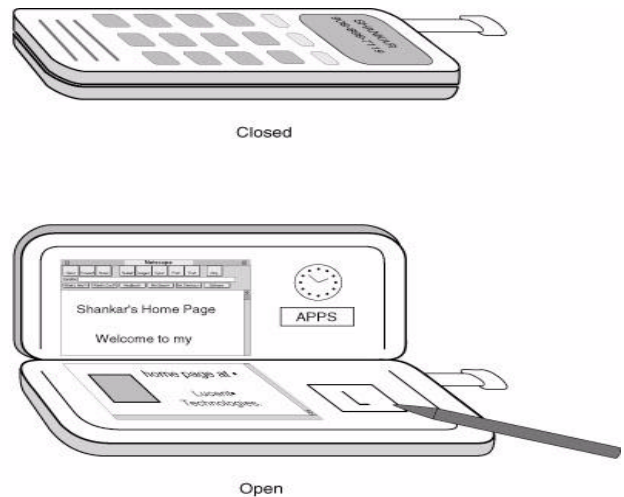


Figure 1: Smart phone hardware design

The pen digitizer is a resistive touch-sensitive membrane laid over the LCD screens. A passive pen is used to tap software buttons, manipulate menus, and write on the touch screen. This is superior to a keyboard, which is thicker, heavier and consumes more power. The small keyboards on PDAs are too small for touch-typing so users hunt-and-peck or use their thumbs. This is error-prone and limits the speed at which text can be entered. A pen doubles as a pointing device, avoiding the need to move the hand from one input device to another.

5. User interface design

There are several components comprising the user interface. Signature verification is used for logging the user into the system. A writing area called the Handwriting Capture Widget (HCW) is used for all handwritten input and corrections. The HCW sends recognized text to applications via Text Entry Widgets (TEWs) which applications use in place of regular entry widgets.

On power up the phone displays the login screen shown in Figure 2. The user prints his name, signs in, then taps “OK” to log in. The screen shot in Figure 3 shows the

phone's look-and-feel after the user is logged in. Only one application is visible at a time. In the figure, the active application is the directory. Handwritten characters are printed in the HCW at the bottom of the screen. The tick-marks on the baseline encourage neater writing and make segmentation and recognition easier.

Menus are placed at the bottom edge of applications rather than the traditional top edge to minimize hand movement and screen obstruction when the user navigates menus. All control buttons and menus are kept at the lower edge or the right edge for this reason. For left handers, the left edge is used rather than the right edge. Control buttons in the HCW are also moved to the left.

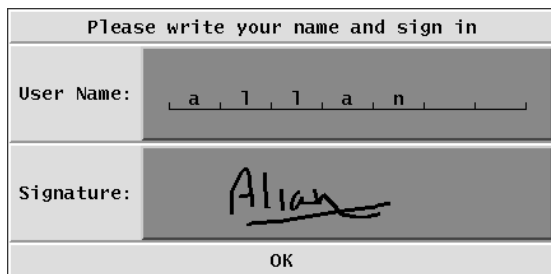


Figure 2: Login screen

5.1. Handwriting capture widget

The HCW encapsulates the handwriting recognizer. The user writes on the baseline and stroke information is sent to the recognizer. Recognized text returned by the recognizer immediately replaces the electronic ink. The last stroke of each character is determined by a time-out and by the position of the next stroke. Longer words may be written by scrolling the writing area with the arrow buttons. "Clear" erases all electronic ink and recognized text from the writing area.

The "Lower" button indicates that the current character set consists of lower-case characters. Tapping on it changes the label to "Upper" and the character set to uppercase characters. Tapping again changes the label to "Digit" indicating numeric characters. Tapping once more resets the button to "Lower". This is necessary to disambiguate certain characters, such as S with s, 0 with o, number 1 with character l. Symbols are active in the vocabulary in all three modes. Tapping "Done" indicates that the user is satisfied with his input, which is then sent to the current TEW.

In case of a recognition error, the user taps on the misrecognized character in the HCW and a menu of the top five recognition results pops up. He then chooses a character from this menu. In the event that the correct character is not in the menu he taps on "Rewrite" and

writes it again. "Start" brings up a menu of available applications.

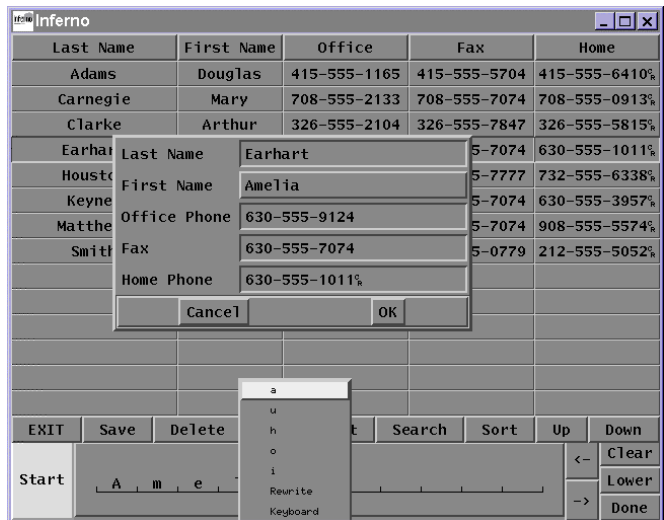


Figure 3: Screen shot with directory active

5.2. Text entry widget

The TEW replaces the entry widget that is used in applications to solicit keyboard input. It communicates with the HCW and its size reflects the smaller size of typed rather than larger handwritten characters. This is especially significant when several TEWs are on the screen at one time, such as entries on a form. Multi-TEW documents also need a mechanism to pass focus from one TEW to the next.

This is done by logically chaining all the TEWs within an application or document. Each TEW knows the ID of a successor TEW which receives focus after the predecessor TEW has obtained text from the HCW. In this way, focus flows linearly within a document without explicit user intervention. The user may override the predefined order by tapping the pen on the desired TEW.

The software library for realizing this chaining function and the transparent connection to the HCW requires very little work on the part of the application programmer. Each application requires one line to initialize the library, one line to register each form and one line per entry widget to declare its position in the focus chain.

6. Handwriting recognizer

There are two conflicting requirements on the ideal recognizer for handwriting input. On the one hand, it should impose as few constraints as possible on the way the user writes; on the other hand it has to achieve an acceptable recognition rate. The recognizers used in most

existing PDAs (e.g., PalmPilot) satisfy the second requirement by sacrificing the first: the user needs to learn a modified alphabet that is often non-intuitive and difficult to remember. Our belief is that given the current state of art for handwriting recognition, we can provide acceptable recognition rates while allowing the user to write much more naturally. The compromise we have chosen is a recognizer for free-style, printed characters: the characters can be written in any natural style, the only constraint on the user is to write each character separately. The recognizer is also writer independent: any user will be able to use it immediately without having to train the system first.

The technology we have developed for such a recognizer is based on statistical models called Hidden Markov Models (HMMs). An HMM describes a doubly stochastic process: a process that generates a sequence of states hidden from observation, and an observable process that is dependent on the underlying state sequence. HMMs have proved to be very successful in modeling speech and on-line handwriting. One of their advantages is that they can model a wide range of variations while still capturing the dominant pattern. In our recognizer, each character is represented by one or more classes depending on the number of distinct styles observed for the character. Each class is modeled by a left-to-right HMM with a variable number of states and discrete state-dependent observation probabilities [4].

During recognition, each handwritten character first undergoes a preprocessing step where a smoothing spline filter is applied to remove noise from the input device and the bounding box of the character is normalized. Then the character is re-sampled at equal-arc-length points to remove original sampling variations caused by different writing speeds, and a set of local shape features (e.g., tangent slope angle, normalized curvature, etc.) are computed at each point. The resulting sequence of feature vectors is then matched against all the character models and the top 5 most likely character candidates are chosen and sent back to the HCW. The matching process is carried out using an efficient N-best decoding algorithm based on the Viterbi algorithm.

The recognizer is trained and evaluated using data distributed by the UNIPEN project [5], which is an international project for on-line handwriting data sharing and recognizer benchmarking. We used release train_r01_v06. The character samples included in this set cover a wide range of writing styles as well as degrees of legibility. Over 50,000 character samples written by 800 different people around the world were used to train the character models. The recognizer was then tested using 22,000 character samples from 600 writers (not included in training). The recognition rates are shown in Table 1.

Table 1: Recognition accuracy

	Top 1	Top 5
Digits	96.8%	100.0%
Upper Case	93.6%	99.4%
Lower Case	85.9%	98.4%

7. Biometric authentication

When the terminal is turned on, the user is prompted to write her user name and to sign in. Signature authentication is the most natural and convenient way for most adults to represent their identities. We have used signature verification as a mode of authentication for a variety of reasons. 1) It provides security to the personal data stored on the device. 2) One does not need to remember passwords and neither can they be stolen. 3) Since we have a pen-interface, no additional hardware is needed to incorporate signature verification. 4) Signatures for identification have been used for a number of years especially in authenticating credit card transactions. Incorporating it in such pda devices helps towards building an infrastructure to authenticate e-commerce transactions performed using this device.

Our scheme of authentication does not just measure the graphical display, but represents an overall graphical expression created through measurable dynamics that attempts to model behavioral tendencies. Unlike handwriting recognition, we are not concerned about the actual letters, language or direction of signing. Many people ignore the letters in their names when they sign; they may even sign as more of a line with a few squiggles. What we do know is that a signature is unique, although it is unique as a behavioral characteristic that can change over time.

Our verification engine consists of pre-processing which includes normalization of signatures using Fourier transform techniques followed by feature extraction. The verifier is based on a comparison of a combination of global and local features of the test and template signatures. The global features capture the overall spatial and temporal characteristics of signatures. The feature statistics of a training set of six genuine signatures are used to build a model or template for validating further test signatures [9]. Some of the global features are the total time taken to sign, the total pen-down time and the average speed of the signing process. A total of 23 global features were extracted from the signatures.

We extract local information about a signature using stroke-direction code (SDC). More specifically, SDC tries to model hand movements that produce a signature and treats them as a time-ordered concatenation of a fixed number of strokes, and derives information about the

spatial orientation of these strokes. The SDC obtained from a test signature is then compared to the model SDC using elastic matching techniques similar to ones used in speech [14]. Our scheme uses adaptive algorithms that give it capability to recognize the changes in individual signatures that occur naturally over time. The algorithms continually update the system's signature model to account for such changes.

This algorithm has a 3% equal error rate tested on a database of 542 genuine signatures and 325 forgeries. Each reference set consisted of the first 6 signatures of every one of the 59 subjects. At the 1% false rejection (FR) point, the false acceptance (FA) rate was about 7.5% and the storage required to store each signature model was about 150 bytes.

8. Implementation

The smart phone uses the Inferno operating system and environment [6]. Inferno is a small, fast, network-centric operating system. It is multithreaded and has built-in support for authentication and security. The Tk widget set [12] is built into the kernel and is used for the graphical user interface.

All of the application and user interface software is written in Inferno's Limbo programming language. The signature verification software and the handwriting recognizer are written in C. The C programs run on a UNIX workstation and communicate with the Limbo programs via TCP/IP sockets to avoid porting existing C programs to Limbo. In a final prototype, the recognizers would be compiled into the Inferno kernel and be accessible as a system library.

9. Future work

Informal testing shows that the use interface minimizes hand movements and works well for this set of applications. Once the software and hardware are integrated, user tests will be performed. There are several questions that need to be answered.

Firstly, what is the maximum acceptable separation between the two screens? The hinge is designed to be very narrow, but it is not known if this is small enough. Secondly, user interactions with applications must be recorded to measure the amount of data going over the wireless network. The network can then be optimized for this traffic. Lastly, what are the minimum terminal size and screen size? But below a certain size, the screen becomes too small and the terminal becomes uncomfortable to hold in the hand.

10. References

- [1] AT&T PocketNet, <http://www.pcsi.com/pal.html>
- [2] D. Cox, "Wireless Personal Communications: A Perspective", in *Mobile Communications Handbook*, J. Gibson ed., CRC Press, Boca Raton, FL, 1996.
- [3] GSM, PCS Access Services Interface Specification in Support of GSM-Based (DCS 1900) PCS Routing Service, Database Services, and Message Transport Service, Bell Communications Research, Red Bank, NJ, 1995.
- [4] J. Hu, S.G. Lim and M.K. Brown, "HMM based writer independent on-line handwritten character and word recognition", 6th IWFHR, Taejon City, Korea, August 1998.
- [5] I. Guyon, L. Schomaker, R. Plamondon, M. Liberman and S. Janet, "UNIPEN Project of On-line Data Exchange and Recognizer Benchmarks", 12th ICPR, pages 29-33, Jerusalem, Israel, October 1994.
- [6] Inferno Home Page, <http://www.lucent.com/inferno>
- [7] IS-136, TDMA Cellular/PCS-Radio Interface-Mobile Station-Base Station Compatibility-Digital Control Channel Addendum No. 1, Electronic Industries Association, Arlington, VA, 1997.
- [8] IS-95, Mobile Station-Base Station Compatibility Standard for Dual-Mode Wideband Spread Spectrum Cellular System, Electronic Industries Association, Arlington, VA, 1995.
- [9] R. S. Kashi, W. Turin and W. Nelson, "On-line Signature Verification Using Stroke Direction Coding", *Optical Engineering Journal*, vol. 35, no. 9, pp. 2526-2533, 1996.
- [10] S. Narayanaswamy et al, Application and Network Support for InfoPad, *IEEE Personal Communications*, vol. 3, no. 2, pp. 4-17, April 1996.
- [11] Nokia, <http://www.nokia.com/com9000/n9000.html>
- [12] J. Ousterhout, *Tcl and the Tk Toolkit*. Addison Wesley, Reading, MA, 1994.
- [13] PDA Buyer's Guide, *Pen Computing Magazine*, vol. 5, no. 20, pp. 94-98, February 1998.
- [14] H. Sakoe and S. Chiba, Dynamic Programming Algorithm Optimization for Spoken Word Recognition, *IEEE Trans. Acoustics, Speech and Signal Processing*, ASSP-26, pp. 43-49, 1978.
- [15] S. Sbihli, Pilot Page, *Pen Computing Magazine*, vol. 5, no. 20, pp. 64, February 1998.
- [16] R. Schneidman, *Wireless Personal Communications*. IEEE Press, 1994.