MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Combadge: A Voice Messaging Device for the Masses

James L. Frankel, Daniel Bromberg

TR2005-161 December 2005

Abstract

While the computer and communication revolutions have had a profound impact on the ways in which we all lead our lives, this impact has not reached much of the world's population. The Combadge project has created a portable device and affordable service that bring person-to-person communication technology to the developing world. This system design allows even illiterate people to exchange messages utilizing easy-to-use devices that communicate our inexpensive heterogeneous networks.

Asian Applied Computing Conference

COMBADGE: A VOICE MESSAGING DEVICE FOR THE MASSES

JAMES L. FRANKEL AND DANIEL BROMBERG

Mitsubishi Electric Research Laboratories, 201 Broadway, Cambridge, MA 02139, USA
E-mail: frankel@merl.com, danielb@alum.mit.edu

While the computer and communication revolutions have had a profound impact on the ways in which we all lead our lives, this impact has not reached much of the world's population. The Combadge[1] project has created a portable device and affordable service that bring person-to-person communication technology to the developing world. This system design allows even illiterate people to exchange messages utilizing easy-to-use devices that communicate over inexpensive heterogeneous networks.

1 Overview/Concept

Dilithium (see figs. 1 and 2) is a small, handheld computing device intended to be a test bed for applications that require neither a keyboard, pointing device, nor display. Combadge is a voice messaging application akin to AOL IM[2], but embedded in an appliance-like device in which audio messages are recorded, transmitted, and played back. In this paper, we will use the term Combadge henceforth to refer to the Combadge application as embedded in the Dilithium device.

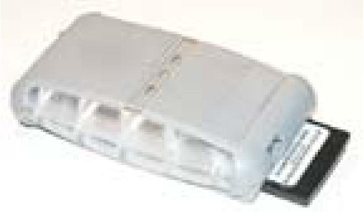

Figure 1. Fully-assembled.

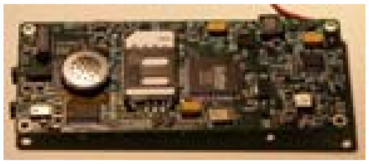

Figure 2. Front side of motherboard.

To demonstrate the use of these devices, imagine a sample interaction using Combadge by one user, Kirk, who wants to send a voice message to another user, Spock. Kirk would push the push-to-talk button on his Combadge and utter "New message for Spock," then release the button. The Combadge would say, "Record your message for Spock." Kirk would then push the button again and utter "We should meet on the bridge in five minutes," then release the button. After each button press, the Combadge will acknowledge with a low-pitched beep and after each button release, the Combadge will acknowledge with a higher-pitched beep. After a brief delay, Spock's Combadge will say, "You have one new message," the five blue LEDs located under each of the translucent left and right side push-to-talk buttons will alternately flash, and the front left green LED will repeatedly flash once separated by a longer pause to indicate one new unheard message. When Spock is able to listen to the message, he presses the button and says, “Play new messages,” then releases the button. Spock’s Combadge replies by playing back Kirk’s recorded message exactly as recorded by Kirk’s Combadge. This is followed by a final low oscillating tone to indicate the end of message playback.

The scenario above is exemplary of the ease of use, simplicity of interface, and accessibility to a broad range of uses. A concise set of commands allows status inquiry and message navigation, as well as the ability to customize the application’s behavior. However, none of these more elaborate commands need be used by the naïve user. This interface makes the Combadge accessible to the illiterate as well as appealing to young children, the elderly, and to those who don’t like the intrusiveness of cellular phones because of the expectation of real-time interaction at indeterminate times (see section 3 below).

One of our goals in the development of this system is to make current communication technology accessible and available to the less privileged and less educated people of the world[3]. To do so, we have created a system in which literacy and wealth are not necessary — we have reduced the cost of the device and the cost of the service over which the device communicates. To demonstrate the effectiveness of such a device, we worked with the TIER Project[4] at the University of California at Berkeley to deploy Combadges in Tamil Nadu, India in collaboration with the MS Swaminathan Research Foundation (MSSRF)[5] during the summer of 2005.

In contrast to voice mail on cellular phones, the system we have designed is simple, optimized for voice messaging, doesn’t require any tree-like menu navigation, and uses voice input and output exclusively. Of course, our software and concepts could be implemented on a cellular phone at the additional cost of that platform. In some sense, the contribution of this project is the aggregation of many concepts into a cleanly designed and easily usable appliance. While somewhat similar to walkie-talkies, Combadge takes the paradigm of rapid exchanges between two remote parties using push-to-talk communication and extends that paradigm by transmitting digitized stored messages to named devices over long distances.

1.1 Earlier Work

Background work includes computer-telephony integration (voice mail[6], Phoneshell[7]), speech recognition (Chatter[8], SpeechActs[9]), standalone handheld speech recognition (VoiceNotes[10]), and one-way voice messaging over pager networks[11]. The wearable computing arena (Nomadic Radio[12]) has touched upon voice-enabled access to services, but not specifically for voice messaging applications. More recent work includes handheld devices requiring constant contact to a high-speed network and server-based speech recognition (SimPhony[13], Vocera Communications[14,15]), PAN-based devices (G-TEK Electronics PWG-300[16]), and numerous VOIP handsets.

This project has extended earlier work by distributing more computation to the handheld device — voice recognition, compression, phonebook maintenance, and message caching are now performed in the device itself — allowing the Combadge to function whether connected or disconnected from the network and over networks of varying bandwidth and latency.

2 Accessibility of Communications Technology

The primary audience for Combadge is the multitudes of people who have not benefited from the digital revolution, thus resulting in the Digital Divide[17]. For much of the world's population, cellular phones and the Internet are still not available or are too expensive. In many cases, a small number of shared phones are used by a large population. In our target rural village communities, many people want to communicate with relatives and friends who have moved to other villages and to cities but illiteracy makes it difficult or impossible.

In constructing an interface that is usable by illiterate people, we focused on voice rather than text as the message content and command dialog. All of the prompts are spoken by the device and all commands are spoken by the user using a simple push-to-talk interface. The device requires little or no training: essentially it has a single push-to-talk button (one on the left for right-handed users and vice versa) and a simple command structure. The voice recognition software is almost completely speaker independent (see sections 3 and 6) and requires no training.

In order to make communications technology accessible to the less affluent, both the device and service costs must be reduced. In most handheld devices, the three most expensive components are the radio transceiver, the processor chip, and the display. In order to reduce the cost of the Dilithium hardware platform, we have designed an interface that neither requires nor includes a display. In addition, we do not use either a keyboard or a pointing device. It is worth noting that the low prices charged for cellular phones do not reflect the cost to produce the device. Cellular phone device costs are recouped over the life of the service contract. When comparing costs, we are comparing the cost of a Dilithium device manufactured in production quantities to the cost of a cellular phone in similar quantities.

Our goal of reaching the third world has required a less expensive service model. We have reduced the operating costs through several means: (1) we utilize a data packet protocol rather than a reserved channel as used by cellular voice communication, (2) we compress all messages prior to transmission, (3) we can use either 802.11b or GSM/GPRS for transmission. Using a data packet protocol does not guarantee real-time transmission of our messages, but this is not an issue because the application is based on queued voice messages. The data packet protocol does guarantee perfect transmission (*i.e.,* data integrity), a feature absent from cellular phones. The compression we perform need not be accomplished in real-time — that is, we can employ better compression at the cost of delaying the transmission.

This compressed message utilizes less transmission time (*i.e.*, less radio spectrum and less power), leaving more spectrum for other users and prolonging battery life. This allows cells to either be larger in size to encompass more devices or allows a higher density of devices per cell. When a local area 802.11b network is available, the Combadge will utilize that network, which is usually free or available at lower cost. Compared to the cellular network, the local area network has higher bandwidth, allows unmetered communication, has lower latency, requires lower transmit power and, therefore, requires less battery power for communication thereby resulting in longer battery life. One important case in which the network would be free is when an 802.11b network is installed for Combadge communication. 802.11b connectivity may also be provided by long distance networks (see section 4.1).

3 Design Issues and Usability

Interaction with the Combadge is structured around the principal method that humans use for communication — speech. This design point results in a device that is approachable and not intimidating. A multi-threaded implementation allows a natural way for the user to control the interaction. By pushing the push-to-talk button and speaking the next input expected by the Combadge, a user can barge in to jump past any phrase currently being spoken. If the push-to-talk button is pressed just for a moment, the current transaction will be canceled. A user might barge in to truncate a familiar long prompt or to cancel sending a message.

Each Combadge supports a phonebook of associations between a familiar nickname and the identity of a Combadge (or, potentially, a grouped list of Combadges). The nickname is called a nametag and is spoken by the user. In addition to running a speaker-independent speech recognition system, each Combadge also runs a speaker-dependent speech recognition system that is used to detect nametags.

As is the case with voice mail and e-mail, the Combadge presents an interface that encourages messages to be answered when appropriate for the recipient rather than when demanded by the sender. This non-interrupting model is often the most suitable mode of interaction.

Much care was invested in the user interface to ensure that the Combadge would act consistently in the presence or absence of a network connection, except for having the capability to transmit or receive messages. All speech recognition is performed within the Combadge and, therefore, dead spots in a network are invisible to a user. Whenever there are new messages and the recipient Combadge is connected to a network, those messages are automatically pushed to the Combadge. Even messages that have already been heard are kept cached within the Combadge unless the memory they use is needed for other incoming or outgoing messages. In addition, new outgoing messages are stored in the Combadge until the Combadge is able to send them to the server. Connectivity status is visible when the user requests to hear a message that is not currently in the Combadge's message cache. In this case, the user is told, “that message is unavailable.” Utility commands also indicate the connectivity state by giving the count of unsent messages (displayed on an LED, too) and the success percentage of pinging the server. If the quality of the connection to a network is poor, the degraded communication channel will increase the time it takes to send or receive messages, but eventually the exact original message will be transmitted. In addition to not requiring a continuous connection to our local network, our system does not require a continuous connection to the Internet either. An Internet connection (or at least a connection to a bridge/router) is only needed when the destination of a message is on a different local area network. This behavior is ideal for a DakNet[18]-like store-and-forward transmission system in the third world in which motorcycles or buses sporadically connect to an 802.11b network and drop off and pick up messages. In addition, several entities[19,20,21] are investigating long-range 802.11 spin-off technologies that would also be suitable networks to use with Combadge.

In principle, our design can lead to increased battery life. As noted above, the Combadge transmits for a shorter time because we can perform better (non-real-time) compression. Also, by sequentially performing audio input, then compression, then transmission, the Combadge reduces its peak power demands. The lack of a display and its backlight also lead to lower power consumption.

Unlike the multitude of voice messaging systems accessible through the PSTN (Public Switched Telephone Network), the Combadge presents a single consistent way to send and receive voice messages to and from any system. Interfaces to voice mail, e-mail, PSTN and other systems (fax, pagers, etc.) are all possible.

4 Description of the Dilithium Device and Operating System

The Dilithium platform is based on the Intel StrongARM processor. It contains 64 Mbytes each of SDRAM and Flash memory. There is an integrated GSM/GPRS modem and on-board SIM for wide-area networking. An optional daughterboard provides one or two Compact Flash (CF) slots for 802.11b local-area networking, additional Flash memory, hardwired Ethernet, or other uses. Two on-board silicon MEMS microphones are used for audio input and can provide active noise cancellation. A relatively wide dynamic-range speaker is used for audio output. The audio CODEC supports many standard sampling rates. The many LEDs on the device offer both visual feedback and visual appeal. The front central LED may be used for input in addition to its usual function as an output[22].

Dilithium includes a two-axis accelerometer that can be used to determine the device’s orientation and movement. This can be used to detect and utilize gestures to control an application. A vibrator is on the motherboard to allow silent operation as necessary. For connection with the outside world, a JTAG connector is provided to initialize the device, a USB connector is provided to allow communication with other USB devices and to recharge Combadge’s battery, extra serial ports are provided to function as a console terminal port and to communicate with other serial devices, two 2.5mm stereo micro phone jacks are available for stereo audio input and output to supplant the built-in microphones and speaker. Finally, there are two push-to-talk momentary pushbuttons on the left and right sides, a power-on button on the top edge, and a reset button recessed on the bottom edge.

The Dilithium hardware required a new boot loader to initialize and test the CPU, Flash and RAM memory, and the peripherals and then to appropriately download and pass parameters to the operating system. Because the Dilithium device is part of a research project with diverse, changing, and expanding goals and is being used by other projects, Linux was chosen as the base operating system. This decision leverages all existing Linux software (including networking code) and all pre-compiled binaries for our CPU architecture. The starting point for the port was Familiar Linux 2.4.19[23]. Once the port was completed, much of Linux was immediately available. Code development used an ARM cross-compiler and remote connection tools such as "ssh."

Linux has provided a multi-language environment in which the Combadge system has been written. C was used for Linux and the Linux device drivers. C++ was used for the speech library. Python was used for much of the application layer. Shell scripts were used for OS interfacing and utility operations.

5 Low-level Combadge Addressing

Each Combadge has a unique address associated with it. The address is the same no matter how the Combadge happens to be connected to the Internet or to other Combadges. Because there are several possible networks through which a Combadge may be connected (802.11b (WiFi), GSM/GPRS, Ethernet, Ethernet over USB, etc.) and because of a desire to enable a Combadge to initially determine its own address without requiring an administrator, a Combadge address is determined based on which hardware is present. Once an address has been determined, the Combadge stores that address in an "identity" file in the Flash memory file system.

6 The Combadge Audio Interface

All Combadge commands are non-modal and expressed in a single utterance (*i.e.*, they don't require navigation through tree-like menus or successive prompting) except for phonebook management. The current audio interface implementation accepts and emits either English or Tamil (selectable via an audio command). The subset of commands that a naïve user might utilize are "New message for <nametag>," "Play new messages," "Play next," "Play previous," and "Reply." A much richer set of commands exists for the advanced user, grouped by voice message sending and receiving, phonebook management, and utility functions.

Speech recognition is performed via a uniform interface named RIST[24] for Reduced Instruction Speech Toolkit. This interface has been implemented to support many underlying speech recognition engines. We are currently using Embedded SpeechWorks SDK[25] for English recognition and SPHINX-II[26] (with substantial modifications to remove floating-point operations) for Tamil.

Any command that may modify the phonebook or the device configuration requires confirmation. The information being confirmed is always repeated back to the user as part of the confirmation request. Command cancellation is always one option given to the user.

6.1 Multi-threaded approach

The Combadge application supports a number of independent activities. In order to eliminate timing dependencies and to simplify the code for these activities, a multi-threaded approach was taken. The user interaction thread implements the command set and blocks only on input from the user. Separate threads deal with long-lived or blocking actions such as speech recognition, message transport, and time-out processing. This design guarantees responsiveness to the user by always being ready to accept a user command. Multi-threading allowed the implementation of a "barge-in" capability. During the execution of a command, a user may press one of the push-to-talk buttons and invoke a new command or skip directly to the next step of a multi-step command (as would be the case for "New message for <nametag>" or "Create contact"), aborting any partially uttered speech output from the previous command. In the degenerate case in which one of the push-to-talk buttons is pressed, but no new command is uttered by the user, any partially uttered speech output from the previous command is aborted and no new command is run. In this degenerate case, if the user is within a multi-step command, the command is cancelled. Similarly, if a user is at an input stage within a multi-step command, the expiration of a timer in a timer thread will cause cancellation of the command.

7 Message Storage and Transport

Each voice message is eventually packaged inside an e-mail message and sent using standard SMTP. Stable storage of all messages ever sent to a Combadge is accomplished using a standard IMAP e-mail system. Each Combadge uses its RAM as a cache for messages. The Combadge can store three classes of messages (listed from most to least important): (1) new outgoing messages dictated by the user but not yet sent to the server, (2) new incoming messages that have not yet been heard, and (3) incoming messages that have been heard but are stored in the Combadge's message cache. Caching messages permits more full-functioned operation when a Combadge is not connected to a network. If the cache becomes full, lower priority class messages are discarded in favor of higher priority class messages.

The current Combadge uses our own connection-based protocol to send messages from a Combadge to a server and to receive messages from a server. A specified server runs a Combadge Daemon program, "cbd." The "cbd" is responsible for communicating with the Combadge. Figure 3 outlines the network infrastructure.

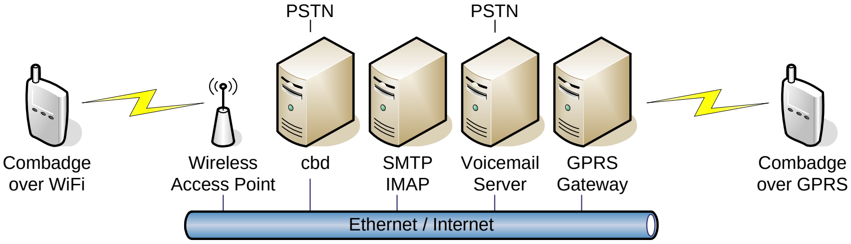

Figure 3. Network infrastructure.

Communication is involved in at least four distinct stages: (1) when a Combadge is powered-on or reconnects after losing contact with the "cbd" server in order to authenticate itself to the server and to participate in an initializing dialog, (2) when a nametag is added to the phonebook in order to validate the Combadge address of the added device, (3) when a voice message is uploaded to the server, and (4) when a voice message is downloaded to the Combadge.

8 Summary

In this paper, we have presented the rationale and design for a voice-messaging device. The device has been fully operational for over a year with a prototype software implementation operational for over two years. The implementation process involved creating new hardware, writing a new boot loader, porting Linux to the device, creating protocols and services, and developing a new application. We have proven the ability to create a useful device that communicates solely through speech. Combadge has been designed to provide handheld communication technology for the developing world and for users who have not currently accepted such technology by providing an implementation adapted to their needs. The result is a simple interface built on a device with low manufacture and service costs. Our future case studies will test the potential deployments.

9 Acknowledgements

The authors would like to thank David Anderson, Ryan Bardsley, Paul Beardsley, Bret Harsham, Mark Haslett and co-workers at Americad, Joe Marks, Barry

Perlman, Divya Ramachandran, Bent Schmidt-Nielsen, Bruce Stetson and the M.A.D.S. staff, and many other Combadge beta testers for their feedback.

References

[1] www.merl.com/projects/ComBadge
[2] www.aim.com
[3] "Tech's Future," Business Week, September 27, 2004
[4] http://tier.cs.berkeley.edu
[5] www.mssrf.org
[6] The 1984 Olympic Message System: A Test of Behavioral Principles of System Design, John D. Gould, *et. al.*, Communications of the ACM, Volume 30, Number 9, Sept. 1987
[7] Multimedia Nomadic Services on Today's Hardware, Chris Schmandt, IEEE Network, September/October 1994
[8] Chatter: A Conversational Learning Speech Interface, Eric Ly and Chris Schmandt, AAAI '94 Spring Symposium on Multi-Media Multi-Modal Systems, March 1994
[9] Talking vs. Taking: Speech Access to Remote Computers, Nicole Yankelovich, CHI '94 Conference Companion, ACM Conf. on Human Factors in Computing Systems, April 1994
[10] VoiceNotes: A Speech Interface for a Hand-Held Voice Notetaker, Lisa J. Stifelman, Barry Arons, Chris Schmandt, Eric A. Hulteen, Proceedings of INTERCHI, April 1993
[11] Voice-Over-FLEX: Value Creation for Paging Carriers and Subscribers, H. B. Choi and Henry Law, IIC Taipei Conference Proceedings, June 1999
[12] Speaking and Listening on the Run: Design for Wearable Audio Computing, Nitin Sawhney and Chris Schmandt, Proceedings of ISWC'98, International Symposium on Wearable Computing, October 1998
[13] SimPhony: a voice communication tool for distributed workgroups, Vidya Lakshmipathy and Chris Schmandt, Video Procs. and Procs. Supplement of UbiComp 2004, Sept. 2004
[14] "Science fiction? Not any more," The Economist Technology Quarterly, Sept. 18, 2004
[15] www.vocera.com
[16] www.gtek.com.tw
[17] Technology and development: The real digital divide, The Economist, March 12, 2005
[18] "DakNet: Rethinking Connectivity in Developing Nations," IEEE Computer Outlook, Jan. 2004 (www.firstmilesolutions.com/DakNet_IEEE_Computer.pdf)
[19] 5G Wireless Communications (www.5gwireless.com)
[20] Efficient MAC Protocol for long distance 802.11 links, Sergiu Nedevschi and Rabin Patra (http://tier.cs.berkeley.edu/docs/projects/wireless_mac.html)
[21] First Mile Solutions, LLC (www.firstmilesolutions.com)
[22] Very Low-Cost Sensing and Communication Using Bidirectional LEDs, P. H. Dietz, W. S. Yerazunis, D. L. Leigh, Intl. Conference on Ubiquitous Computing (UbiComp), Oct. 2003
[23] familiar.handhelds.org
[24] RIST is a simplified speech recognition interface implemented at Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA, USA
[25] SpeechWorks now part of ScanSoft, Inc. (www.scansoft.com/speechworks)
[26] The SPHINX-II Speech Recognition System: An Overview, Xuedong Huang, *et. al.*, Computer Speech and Language, 7(2), pp. 137-148, 1992