ListenIN: Ambient Auditory Awareness at Remote Places

by

Gerardo Macario Vallejo Rosas
B.S., Electrical Engineering
Instituto Tecnologico y de Estudios Superiores de Monterrey, 1998

Submitted to the Program in Media Arts and Sciences, School of Architecture and Planning, in partial fulfillment of the requirements for the degree of

Master of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2003

Author

Gerardo Macario Vallejo Rosas
Program in Media Arts and Sciences
August 15, 2003

Certified by

Christopher Schmandt
Principal Research Scientist
Speech Interface Group, MIT Media Lab
Thesis Supervisor

Accepted by

Dr. Andrew B. Lippman
Chair, Departmental Committee on Graduate Students
Program in Media Arts and Sciences

ListenIN: Ambient Auditory Awareness at Remote Places

by

Gerardo Macario Vallejo Rosas

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning,
on August 15, 2003, in partial fulfillment of the
requirements for the degree of Master of Science

Abstract

In this thesis we propose the architecture of a modular system that allows closely related people, such as relatives and friends, to maintain awareness of a distant site, such as home, through auditory cues and eavesdropping.

While people are at work or traveling, they often want to have some degree of awarereness about activities related to family members' health and well-being. The solution that we propose is a light infrastructure system that listens into the home and in response to a change in activity (e.g. cooking, taking a shower, watching TV) sends a few seconds of audio to the caregivers providing them with background awareness of events occurring at the remote site. In order to protect privacy at the monitored site the system is designed to classify sounds and then send audio icons, pre-recorded pieces of sound representing activities (e.g. a recording of a baby crying), instead of the actual sound. However, when the classifier's confidence level is low and speech is detected, the system sends the actual sound but in a garbled way. In the absence of speech, the system simply sends the actual sound. The system takes classifying decisions based on acoustic evidence and possible location of the source. Several clients can connect to the system and simultaneously receive alerts.

Thesis Supervisor: Christopher Schmandt

Title: Principal Research Scientist, MIT

Thesis Committee

Thesis Supervisor

Christopher Schmandt
Principal Research Scientist
MIT Media Lab

Thesis Reader

V. Michael Bove Jr.
Principal Research Scientist
MIT Media Lab

Thesis Reader

Kent Larson
Principal Research Scientist
MIT Media Lab

Thesis Reader

Lorin Wilde
Research Affiliate, Lecturer
MIT Research Laboratory of Electronics,
MIT Media Arts and Sciences

Acknowledgements

Being at the MIT Media Lab is an experience that I will never forget. I have learned that I know very little about this world and that we are only humans trying to persist in human's memories. I cannot help but thank:

Chris Schmandt, my advisor, for opening the doors of this place to me, guiding me through this journey and being a nice human being.

My father. I could write an entire book about how thankful I feel about you but what I can shortly say is thank you for being my unconditional friend and guide during all my life and for believing that I am greater than I really am.

My mother for teaching me the basics of life, for putting all the passion that a brave woman has into raising good human beings, for teaching me how to fight until the last breathe. I miss you.

Adriana for encouraging me to take this opportunity, for teaching me what life is really about, for being my partner and, for helping me to realize how small I am without you. I love you.

Marcela and Xochitl, my sisters, for being a strange mix between friends and replacement mothers. I love you girls.

Emmanuel for giving me the blue pill that released me from The Matrix. Thanks for listening and being a real friend for the last two years.

Wilfrido for being my friend, officemate and lunch partner.

Natalia for your friendship and fine advice, for being an extraordinary human being and teaching me to not give up under any circumstances.

Tanzeem, for lending me some wearable computers.

Speech Grads gang for making me feel comfortable during my stay in the group.

Table of Contents

List of Figures

Chapter 1

Introduction

Today's jobs are becoming more and more demanding. People spend much of their time working and traveling, reducing the time they can invest to take care of individuals who depend on them.

The telephone is now a ubiquitous medium and it is very easy to place a phone call from almost any city in the world to virtually anywhere and stay aware of relatives and friends' lives. However, a telephone channel requires that at least one party initiate the call and also demands significant attention from both parties [2].

We believe that most caregivers would like to have some hints about activity at remote places in order to have peace of mind [7] and they want that information presented at their location in a non-distractive manner. These activities provide caregivers with valuable information about the well-being of their relatives. Nevertheless, individuals occupying those remote places require some degree of privacy. Obviously, there is a trade-off among several factors that must be fulfilled in order to solve the problem of maintaining awareness at remote places.

With the expansion of the Internet, people now can have open chat-text channels to stay in touch such as ("I seek you") ICQ system [13], which indicates a person's availability through labels such as away, busy or available. Moreover, the Internet also offers Voice over Internet Protocol (VoIP), which has emerged as a low-cost alternative to regular telephone service. However there are practical problems in using existing

technology; obtaining feedback by caregivers requires that their relatives interact with Internet applications. Using an unfamiliar technology can be intimidating for seniors and children. On the other hand, the Internet seems to be a suitable platform to develop new applications to maintain awareness of remote places. Computers with Internet connectivity are almost at every work environment and each day more and more handheld devices are getting Internet connections, such devices could easily subscribe to systems that maintain awareness at remote places.

In this thesis we propose a system that allows closely related people, such as relatives and friends, to maintain awareness of a distant site such as home through auditory cues and eavesdropping. By using Internet connections activity information can be presented as an audio icon, a pre-recorded piece of sound representing a particular activity (e.g. a recording of a baby crying), instead of the actual sound, on a desktop or any mobile device having an Internet connection.

ListenIN uses a novel architecture for distributed audio sensors. ListenIN attempts to classify activity at a monitored home environment based on acoustic evidence collected from these sensors. When a transition in activity is successfully recognized an audio icon is sent to the Client Interface to inform the caregiver about the event. However, when the classifier's confidence level is low and speech is detected, the system sends the actual sound but in a garbled way. In the absence of speech the system simply sends the actual sound.

It is important to remark that ListenIN does not contribute to solving the problem of accurate activity recognition using pattern recognition techniques. ListenIN can be considered as a new attempt to maintain significant awareness for caregivers to infer that the person being monitored is generally engaged in normal activity. The novel architecture of ListenIN allows for an easy integration of different kinds of sound classifiers and has no intention of replacing social communication; it is intended to be a tool to help caregivers know dependent's urgency of human contact or assistance.

1.1 An Example Scenario

To better understand how ListenIN can help caregivers and dependents we present the following scenario.

John is a widower in his mid-seventies. John has a son, David, who works for a company as an engineer. John wants to remain in his apartment however he is experiencing some health problems that are threatening his independence. David as John's only son feels responsible for taking care of his aged father. ListenIN running at his father's home alerts him about activities occurring there. For example, when John wakes up in the morning and goes to the bathroom ListenIN attempts to recognize activities such as brushing teeth and flushing the toilet. Thus, it sends alerts to David's computer. At David's office, his computer gets the alerts and plays back the corresponding audio icons. David hears the audio icons and is aware that his father's day started as usual.

Later on in the afternoon Bob, John's brother, visits him and they start chatting while drinking coffee in the living room. ListenIN attempts to classify activity detected in that room of the house but it fails since it is difficult even for humans to classify such an unconstrained activity. However, ListenIN detects that speech is present in the room audio and, in order to protect privacy, it randomizes 100-millisecond blocks of the audio to make this speech content less intelligible. ListenIN is now ready to send five seconds of the resultant audio to David, the caregiver. Even though David is not able to understand the garbled speech sent to his computer, he is still able to recognize that Bob is there and David can infer that his father has company and that everything seems to be normal. The information provided by ListenIN provides awareness of his father's well being, giving David peace of mind.

1.2 Contributions

Existing systems that attempt to maintain awareness at remote places require complicated setups at the monitored side such as installing multiple cameras, wiring sensors or, even

worse they require that monitored individuals (who in this thesis we call subjects) wear sensors. Such systems are not portable and they interfere is inhabitant's daily life.

ListenIN is an effort to overcome the complexity that usually keeps a system from being portable. There is obviously a trade-off between the complexity of setting up a system and the amount of information it can capture and process. ListenIN hypothesizes that enough information to classify activity can be extracted from audio signals alone. Moreover, audio by itself is an attractive medium due to its low processing complexity and cost compared to other media.

Some work has been done using audio to represent recognized activity. ListenIN implements this idea by delivering an audio icon, a pre-recorded piece of sound representing a particular activity (e.g. a recording of a baby crying), instead of the actual sound. At the caregiver's side, an audio icon conveys enough information to associate it with a particular activity while maintaining privacy at the monitored side.

Finally part of the outcome of this research is a robust algorithm to discriminate speech from non-speech sound in noisy environments. The ListenIN architecture requires such a discriminator. Originally we planned to use common Digital Signal Processing techniques to implement it. However, we realized that existing techniques were designed assuming a Signal to Noise Ratio (SNR) that by far exceeds real conditions in a home environment. The algorithm used by ListenIN is described in section 4.3.2.

1.3 Document Organization

The rest of the document is organized as follows: Chapter Two presents some previous and related work that has motivated the development of ListenIN. Chapter Three describes the architecture of the server that supports the activity classification task in ListenIN. Chapter Four describes the technical details of the implementation of each module that integrates the architecture of the system. Chapter Five describes deployment and evaluations of the user interface, the system architecture, and the system performance of ListenIN. Finally, Chapter Six presents contributions and future work that may be done to improve performance of ListenIN.

Chapter 2

Background

In this chapter we describe some previous and related work that has motivated the designing of features and development of ListenIN.

For the purposes of this thesis the term "awareness", refers to having knowledge to draw inferences about activities at remote places.

Systems that attempt to maintain awareness usually require complicated setups such as installing multiple cameras and wiring sensors besides they usually use more than one PC to run heavy recognition algorithms. ListenIN is designed as a new attempt to overcome complexity that keeps systems from being portable and makes them unfriendly and intimidating to monitored people (who in this thesis are called subjects). In terms of user interface, the part of the system that presents caregivers information, ListenIN is designed to maintain constant but minimally intrusive awareness information about activities taking place at remote places.

Existing related work includes systems that maintain activity awareness at remote places and use explicit representations to present information at remote sites. Since most of these systems were designed to connect distributed workgroups they may not consider privacy an issue. Explicit representations such as raw video and audio can be harmful for privacy at the monitored site. Basically everything said or done at one location is transparent to the others.

A different approach that tries to minimize the privacy problem consists of presenting abstract representations of recognized activity at remote places. Visual media have been widely used but little work has been done using audio to represent recognized activity at remote places. We find audio an interesting medium to convey information that provides background awareness about activities occurring at remote places.

Some related work has been done using audio-only interfaces. Inside this cluster we find systems that try to connect working groups at different physical locations by providing acoustic awareness information. Audio interfaces have demonstrated effectiveness to display information to users while being minimally intrusive. However little work has been done using this kind of interface to connect people in environments other than work.

In the following sections we describe in a more detailed way the clusters of systems introduced above along with a brief explanation about how ListenIN differs from existing work.

The first two sections describe related work relevant to the design of the user interface. The third section presents previous work done to increase accuracy of the recognition task at the cost of installing complex and sometimes intimidating systems. The fourth section describes efforts to overcome complexity. The next section describes research using audio-only interfaces. Finally the last section gives an overview of privacy considerations when designing systems that monitor human activity.

2.1 Systems that use explicit activity representations

Several existing approaches to maintain awareness of other people's activities use explicit representations to present information at remote sites. Among these approaches we find systems like Portholes [23] and Montage [14]. Portholes provides group and individual information to members of a physically distributed work group. This system provides an integrative view of one's community through a matrix of still video images. These images are snapped every five minutes and the matrix is updated automatically. As a result, users can get a peripheral or background sense of their co-workers and activities.

The Montage system provides lightweight audio-video glances among distributed collaborators and also integrates other applications for coordinating future contact.

Each of these interfaces is a desktop conferencing tool designed for distributed workgroups. Both provide computer-mediated communication through computer networks. However, these systems require explicit user interaction with a desktop. Privacy is not considered an issue since both systems were designed for work environments.

2.2 Systems that use abstract activity representations

On the other hand, interfaces that use abstract representations have been designed to minimize the problem of invading privacy of the observed party. Digital Family Portraits [7] is a system representative of this cluster of interfaces. The system monitors and reports the level of daily activity of an elderly parent to a trusted group in minimally invasive and private way. Digital Family Portraits provides qualitative abstract visualizations of a family member's daily life from available sensor information.
Another system that uses abstract representations is Video Finger [17]. Video Finger uses previously recorded movie shots of community members to represent their presence in a LAN. Video Finger represents a login to the network by playing the movie of that particular user entering to a room and taking seat. Logout is represented by playing the movie shot of that user leaving the room.

Little work has been done using audio to represent recognized activity. ListenIN implements this idea by delivering an audio icon, a pre-recorded piece of sound representing a particular activity (e.g. a recording of a baby crying), instead of the actual sound. At the caregiver's side, an audio icon conveys enough information to associate it with a particular activity while maintaining privacy at the monitored side.

Speech is a special and extremely important audio signal under the architecture of ListenIN due to privacy issues. ListenIN is always looking for presence of speech and when detected, again in order to protect privacy, it randomizes one hundred millisecond

blocks in the original audio to make it less intelligible; this modified audio is sent to caregivers.

2.3 Complex Infrastructure Systems

Complex infrastructure systems have been used for recognizing daily activities. For example, computer vision sensing for tracking [26, 11, 27] subjects and action identification [29, 15, 4] often works in the laboratory but fails when deployed at home environments due to every day factors, such as, clutter, variability in lighting conditions, and unconstrained number of different activities.

Audio is less sensitive to physical changes in the environment than other available signals. ListenIN retrieves information from audio and attempts to automatically classify activities based on acoustic evidence.

Attaching sensors to the body is another way to get data from subjects. From the design point of view this approach is not an expensive solution and designers can retrieve valuable data. For example, an accelerometer, an electronic transducer that converts acceleration to an electrical signal, can be used as a motion sensor [10, 20, 16].

ListenIN does not require subjects to wear sensors instead it uses a novel architecture that gathers data from distributed audio sensors.

2.4 Light infrastructure Systems

Several efforts have been done to relieve homes and subjects from heavy infrastructure such as wiring cameras or having subjects wear small computers.

E-Care Assistant [6] is a commercial system that avoids the use of cameras, microphones, special wiring, or computers at the monitored site. Motion sensors and a base station are the two basic components of the system that live in the monitored home. The collected information is transmitted to a remote server operated by the company that sells the system. Motion sensors track movement within the home. The movement is then transmitted over the senior's existing telephone line to the company's server. There it is analyzed using algorithms that look for deviations from the senior's "normal" behavior. For example, if Mom doesn't get out of bed one morning, a message is sent to the

caregiver, "Check on Mom." Since the system is IP based, the message is sent automatically to e-mail addresses, cell phones, pagers etc. In addition, the system uses a web-based interface that allows caregivers to log on to find out about the safety of their loved one. The page shows the senior's safety status for key activities of daily living, such as rising from bed in the morning to use of medication, using the bathroom, or kitchen facilities.

Low cost open-close sensors have also been used to retrieve data from subjects [9]. The sensors are installed in strategic home appliances such as, drawers, light switches, stove, etc. to collect data from subjects while performing daily activities. The data is processed offline to classify activity based mostly in time patterns. Since this system was designed to detect small changes in daily behavior that could be used for proactive management of healthy behaviors, the information granularity obtained exceeds by far the requirements of a system that attempts to maintain significant awareness for caregivers to infer that the person being monitored is generally engaged in normal activity.

The architecture of ListenIN uses a light infrastructure as well. The classifying of activity relies on correct classification of individual sounds. Location of the sound source is a feature trivial to extract but important for improving accuracy. For example, without additional information the sound of water running might increase error probability between classifying the activity as washing dishes and washing hands. If the system has evidence that the sound was produced in the bathroom, it is more likely that the system can correctly classify the activity as washing hands. In summary, by incorporating location information we can easily disambiguate similar sounds.

The architecture of ListenIN utilizes sensors to detect localized audio and retrieve location information that aids the classification process while maintaining a lightweight infrastructure.

2.5 Audio-only Systems

A different approach to the problem of maintaining activity awareness at remote places consists of using an audio-only interface. An audio-only interface, as its name suggests,

only takes audio as both input and output. Several studies have been done to demonstrate the effectiveness of audio as medium to convey information [5, 28].

Among these systems is Thunderwire [19]. Thunderwire is conceptually similar to a telephone conference call. It allows any number of group members distributed at different locations, to be simultaneously connected and anything said at any time by any member is heard by all.

Thunderwire maintains a full duplex high-quality audio channel between two groups. Basically, this system is an always-on, audio-only interface that does not take into account user's activity to preserve privacy.

The following are the characteristics of Thunderwire:

- Thunderwire is a purely audio medium. Except for an "on" light, it has no other visual interface or cues.
- It uses high quality audio, such that users can easily distinguish one another's voices as well as overhear background sounds. The sound quality makes it possible to hear everything one might hear sitting in a person's office, including private vocalizations, phone calls and bodily and background noises.
- The act of connecting or disconnecting is indicated only by a barely audible click. In fact, there is no way to know exactly who is listening without asking.

Thunderwire can be considered as an obvious solution to the problem of remotely monitoring a home environment. However, such an interface is extremely intrusive at the caregiver's side, does not have memory of previous events and uses bandwidth in an inefficient way.

A similar approach is Family Intercom [18]. The inter-home communication modality of this system is a context-aware intercom that is not always on; the connection persists for the duration of the conversation between two remote users (at different places), which can be terminated as the result of context related activities. The monitoring of activities at a remote place is only done under explicit request from the caregivers. The main disadvantage of this system is that caregivers may miss some important cues related to their relatives well being.

ListenIN attempts to maintain constant but minimally intrusive awareness of remote places. ListenIN is not an always-on interface: it transmits information to caregivers only

when transitions in the average energy of sound in a particular room of the monitored home are detected, such transitions are taken as transitions in activity. This way, it is designed to convey valuable information to caregivers while minimizing interruptions.

2.6 Privacy Issues when monitoring activity

ListenIN addresses the following points in subject's perception about privacy [24]:

a) Information sensitivity. This relates to subject's perception about the sharing of information. The architecture of ListenIN allows that audio sensors can easily be removed from areas when subject's sensibility about privacy is jeopardized.

b) Information usage. The manner in which collected information will bring benefits. ListenIN is currently targeted toward a care-giving situation, in which the monitoring party has special responsibility, either implicit (aging parents) or explicit (children under someone else's care). Subjects (party being monitored) obtain the benefit of living independently (aging parents case) while being taken care of remotely. There is always a trade-off between privacy and level of help that both subjects and caregivers can obtain from the monitoring system.

Any system that claims to present awareness information is inherently suspect from a privacy standpoint [25], and this concern is amplified when systems are deployed in domestic environments. While we are concerned with privacy issues, this thesis intentionally addresses only a portion of them, as it is a tentative foray into the area. Factors we are explicitly not yet addressing are local awareness of monitoring status, symmetry of awareness information ("I hear you means you hear me"), local ability to override audio input, and the broader picture of how such awareness indications may be built into communications technology which will better serve both parties.

However, ListenIN is not privacy-naive. As with other systems in which processing is interposed between sensors and human recipients, the information to be accessed can be constrained, masked, or abstracted to reveal less about the person being monitored. ListenIN would ideally recognize every sound produced, map this to a particular activity, and send an audio icon and never any acoustic data from the home. Since this is only partially feasible currently, an intermediate position is to detect the presence of speech

and garble the audio so as to render the speech unintelligible, just as in [12]. Finally, even if the actual sound is passed out of the home unmodified, it is limited in duration by the server on premises. Of course with any such technology intervention, unless the residents trust the technology (there are various methods to raise trust, such as allowing them to hear what is actually going out) they may not trust this processing.

It is easy to have knee-jerk pro-privacy reactions. This thesis work is, however, informed by experiences of both the author and his academic advisor in caring for aging parents remotely. As parents age, they become less reliable self-reporters, and yet need ever-increasing assistance. It is admittedly difficult to negotiate informed consent and acceptance that others are increasingly responsible for one's care, especially as the elderly lose mental ability. Listening in, much as one might do through the walls of an "in-law" apartment, may be minor compared to the personal intimacy of dealing with medical or financial problems at this period in life.

Chapter 3

Server Architecture

This chapter describes the architecture of the server that supports the activity classification task. Implementation of the ListenIN architecture will be discussed in the next chapter.

The ListenIN system has two main parts: 1) The Client part that lives at the caregiver's side, which consists of a thin audio interface that informs about activity occurring at the monitored home and 2) the server part that lives at the monitored side, which handless all of the classification tasks and dispatches information to caregivers (Figure 3-1). At the server side several Detection Modules detect loud localized audio and pass this information to the Sound Collecting Module installed in the same room as the Detection Modules. Finally, the Sound Collecting Module sends the passed information along with room audio to the Classification Module, which is implemented in a PC, and where activity classification is performed.

This chapter describes the architecture at the server side; which represents the core of the ListenIN system. The first section describes the ListenIN architecture. The rest of the sections describe the modules that integrate the architecture.

3.1 Architecture Description

ListenIN presents an architecture that consists of three modules depicted in figure 3-1: (1) Detection Module (DM), (2) Sound Colleting Module (SCM) and (3) Classification Module (CM).

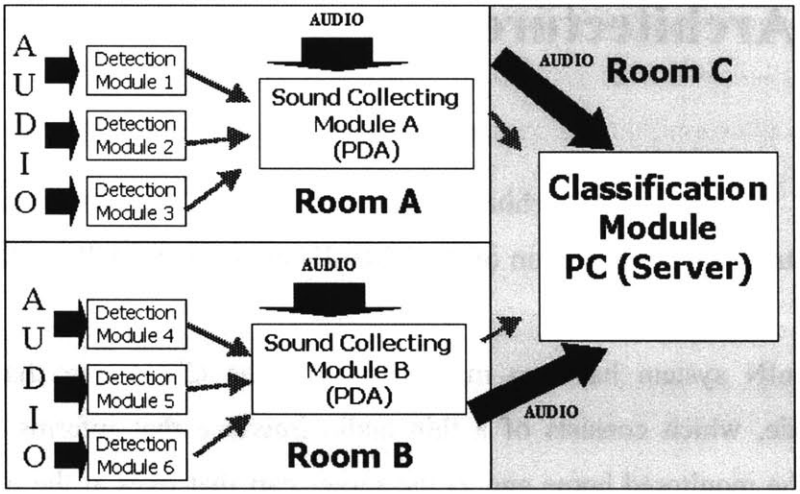

Figure 3-1. The ListenIN Architecture. Thick black arrows represent audio. Thin arrows represent data

Figure 3-1 shows the ListenIN architecture. The thick black arrows represent audio and the thin black arrows represent data. The lowest node in the ListenIN architecture is the Detection Module (DM). There are several Detection Modules per room sending data to their corresponding Sound Collecting Module (SCM). The DM's are basically noise detectors dedicated to provide the Classification Module (CM) information about the location of the sound source, information that aids the classification process.

The SCM, which is implemented on a PDA, communicates with the DM's through a Master-Slave wireless communication protocol. The SCM is physically close to DM's assigned to it; this is located in the same room. The SCM takes the location information coming from the DM's and sends it along with room audio that the SCM directly captures through its microphone using 802.11b networking.

Finally in the top of the hierarchy is the Classification Module (CM). The Classification Module is implemented on a PC Pentium IV and is in charge of classifying inhabitants' activities and detecting presence of speech based on acoustic evidence collected by the previous modules.

3.2 Detection Modules (DM's)

The Detection Modules (DM's) are low powered RF audio sensors, dedicated to detect loud localized audio. These sensors are placed on possible sound sources that can give the CM cues about local activities, for example, sound from the TV in the living room, noise from the stove in the kitchen, or noise from the toilet in the bathroom. The complexity of these sensors is very low. Each sensor consists of a unidirectional microphone connected to a microprocessor and has a binary output. Basically, the sensor is looking for an abrupt change in the energy of the sound within a range of a few centimeters. When the sensed energy goes beyond a threshold, the sensor changes its output indicating that it has detected a possible source of activity. This binary "noise-detected/not detected" flag is transmitted via low power RF to the SCM and eventually to the CM. Since the CM keeps in a table all the DM's and their location (see Table 3-1), it can use this flag to know the location of the sound and use it as an extra feature in the classification process to disambiguate similar sounds.

The algorithm used by the DM's to sense changes in audio energy is a variant of the one used in [2].

3.3 Sound Collecting Module (SCM)

The Sound Collecting Module (SCM) is a wireless microphone transmitting audio directly to the CM using 802.11b networking. This module also communicates with several DM's planted in the same room using a low power RF transceiver.

The transmitting of audio to the CM is done in response to a detected change in the average energy of the sound room, which is taken as a change in room activity. The transmitted audio has two parts: 1) continuously transmitted raw audio until the SCM receives the order to stop from the CM and 2) a piece of raw audio corresponding to one

second before and nine seconds after the transition in activity is detected. This is the piece of audio that is sent to the Clients when all attempts to classify activity fail. By recording one second before the transition is detected, users are provided with acoustic context that can be used to figure out what originated the sound.

As mentioned before, there is only one SCM per room transmitting audio and the "noise-detected/not detected" flags collected from several DM's to the CM. The protocol used between the SCM and the DM's is a RF Master-Slave configuration.

Since there are several SCM's transmitting to the CM we need a way of controlling them and avoid collisions. One solution is to use Time Division Multiplexing (TDM) where each microphone is given a time slot to transmit. The implemented solution was to give each SCM an IP address and use TCP socket connections to safely transmit audio. However this solution requires a developing platform that has a sound device and that also supports wireless TCP/IP connectivity (802.11b). Having these requirements we use a PDA to implement the SCM.

3.4 Classification Module (CM)

The Classification Module (CM) takes audio and source location from previous modules using 802.11b networking and based on this information it attempts to automatically classify activities. To implement this module we use a PC, Pentium IV since classification is a process that requires significant computational power. The Classification Module has two main components: 1) the Activity Classifier (AC), that is the core of the activity classification engine and 2) the Speech Detector (SD).

Activity Classification in ListenIN is done using three techniques: 1) Sound Classification using pattern recognition techniques (sound similarity), 2) Activity Classification based on location of the sound source inside the house and its duration and 3) Sound Classification based on energy bursts detection and their temporal characteristics.

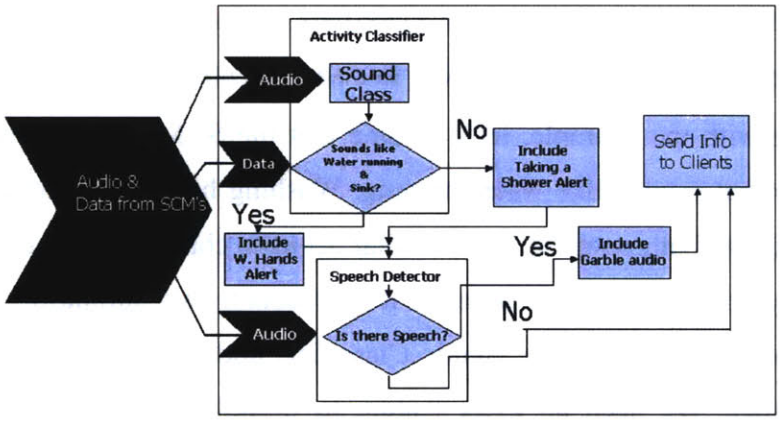

Figure 3-2. The Classification Module. The CM receives both audio and data from multiple SCM's

To better illustrate what the CM does, Figure 3-2 shows a flow chart of the audio and data processing in the CM for activity taking place in bathroom. In this case the Activity Classifier (AC) classifies activity based on sound similarity.

In the left side of the figure, we have both audio and data as an input to the module. Inside the CM audio and data are separated. The AC processes the incoming piece of audio running it through each of the sound classifiers, which extract acoustic features and deliver similarity measures.

The AC stores location of the SCM's and their corresponding DM's in a data table similar to the one shown in Table 3-1.

SCM#	DM#	Room	Location
1	1	Kitchen	Stove
1	2	Kitchen	Table
1	3	Kitchen	Sink
1	4	Kitchen	Oven
2	1	Bathroom	Sink
2	2	Bathroom	Toilet
2	3	Bathroom	Shower
2	4	Bathroom	N/A

Table 3-1 SCM's and DM's location table

Since in a home environment there are sounds that have acoustic characteristics similar enough to confuse even human classifiers, the AC relies on collected "noise-detected/not detected" flags from the DM's to disambiguate similar sounds.

Continuing with the example, let's assume that the SCM installed in the bathroom captures audio that sounds like water running. Washing hands and taking a shower are two activities in the bathroom that have similar acoustic characteristics to water running. This obviously confuses the AC because it will obtain two similarities very close in magnitude: one for the Washing Hands Classifier, and the second for the Taking a Shower Classifier. We need an extra feature to disambiguate activities that "sound" similar. The label "data" in Figure 3-2 refers to the "noise-detected/not detected" flags collected from DM's. The AC uses these flags as the needed extra feature to aid the classification process as shown in the flow chart. We observe that the if the AC gets a "noise-detected" flag from the DM located in the sink, then it will classify the activity as Washing Hands, resolving ambiguity. In this case the CM would then prepares the Washing Hands alert to be sent to the caregivers.

Once the AC has a valid output the CM tries to find out speech in the processed audio through the Speech Detector (SD). The SD is a speech/non speech discriminator that implements the algorithm in [1]. If speech is found, the CM, in order to protect privacy and using the algorithm proposed in [1] randomizes 100-millisecond blocks in the original speech to make it less intelligible. In this example we assume that there is no speech in the room audio so the CM is ready to send activity information to the caregivers.

For this example ListenIN only sends the Washing Hands Alert. At the caregiver's side, the client receives the command to play the corresponding audio icon.

Chapter 4

Implementation

This chapter describes the technical details of the implementation of each module that integrates the architecture of ListenIN. As described in the previous chapter there are three modules in the architecture: 1) The Detection Module (DM), 2) The Sound Collecting Module (SCM) and 3) The Classification Module (CM).

It is important to point out that the work here shown is not intended to solve the problem of accurate ambient audio classification; instead, it uses existing algorithms to perform this task.

The core of the Detection Module is implemented on a hoarder board, a wearable data acquisition board, designed by the wearable computing group at the Media Lab [32, 31]. The programming of these devices is done using a native C complier for the PIC micro controllers family. The wireless link between the DM and the SCM uses a low power RF transceiver implemented in a separated board that we call Transmitting Board (TB). The TB is used in both the DM and the SCM and is controlled through serial port (RS-232).

To implement the SCM we use a PDA Compaq Ipaq H3600 running Linux OS. The software running in the SCM is written in C/C++. The SCM uses an 802.11b wireless card to transmit audio and data to the CM. To communicate with the DM's the SCM uses the TB.

The last module, the CM, is implemented on a PC Pentium IV running Windows 2000 OS. This software is written in C/C++. The sound classification and the Speech

Detector algorithms are implemented in Matlab for fast prototyping. The CM uses a C++ API developed by MathWorks [21] to communicate with the Matlab engine and have access to the classifiers at run time.

The first section describes the implementation of the Detection Module and its noise detection algorithm. The next section describes the software implementation of the Sound Collecting Module and finally the last section shows how the Activity Classifier and the Speech Detector algorithms work and how they are implemented within the Classification Module.

4.1 Detection Module (DM)

The DM (Figure 4-1) is implemented on a hoarder board, a low cost wearable computer designed at the Media Lab. The software used in this module was written in native C for the PIC microcontrollers family.

4.1.1 DM Hardware Implementation

The DM is implemented on a hoarder board. It uses a low gain unidirectional microphone to detect audio only within a radius of few centimeters.

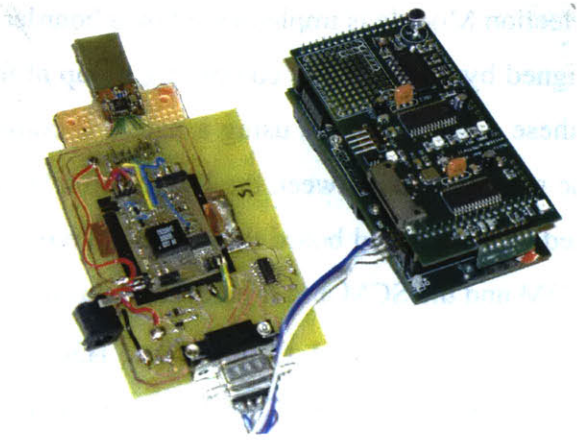

Figure 4-1From Left to right Transmitting Board (TB) connected to the Hoarder Board, these two devices integrate the Detection Module (DM)

As shown in figure 4-1, the hoarder board connects through its serial port to the Transmitting Board (TB) to communicate with the SCM. The TB is basically a RF transceiver controlled by a PIC16F628. The transceiver used in the TB is the Nordic nRF2401 [22], a low power, low cost, RF transceiver that has Cyclic Redundant Check (CRC) hardware, a standard method for error checking and correction in data communication and First Input First Output (FIFO) buffers for both Transmitting and Receiving.

The output of the DM is a binary flag: TRUE for noise detected, and FALSE is the default state, means no noise detected.

4.1.2 DM Software Implementation

As mentioned in the previous chapter, the DM's are dedicated to detect loud localized audio. These modules communicate with the SCM in a Master-Slave configuration in which the SCM has the Master roll. In other words, the SCM is continuously asking each DM for the instantaneous status of the noise detector.

Under the ListenIN architecture the DM is continuously monitoring the air looking for requests from the SCM about the status of the noise detector. When a new request comes in, the software in the TB sends an interrupt to the hoarder board and asks for the instantaneous status of the noise detector (TRUE or FALSE). The hoarder board replies and the TB takes this binary flag and sends it to the SCM. Finally the TB returns to its idle state. Appendix A contains the source code for the implementation of the TB and for the hoarder board.

The code shown in Appendix A is written for a native C PIC compiler called CCS. This piece of code works for both Master and Slave configurations. To compile the code for Slave configuration, the directive SLAVE should be defined.

There are three important functions in this code. The first one is `listen_to_channel(int chan)`, which makes the TB monitor for incoming request from the Master (SCM). The second function is `ask_HB_for_peak()`, which interrupts the hoarder board and asks for the instantaneous status of the noise detector (TRUE or FALSE). And finally `tx_peak_val()` , which commands the TB to send the noise detector status to the SCM.

Noise Detection Algorithm

The algorithm used by the DM's to sense changes in audio energy is a variant of the one used in [2] for the Cry Baby project.

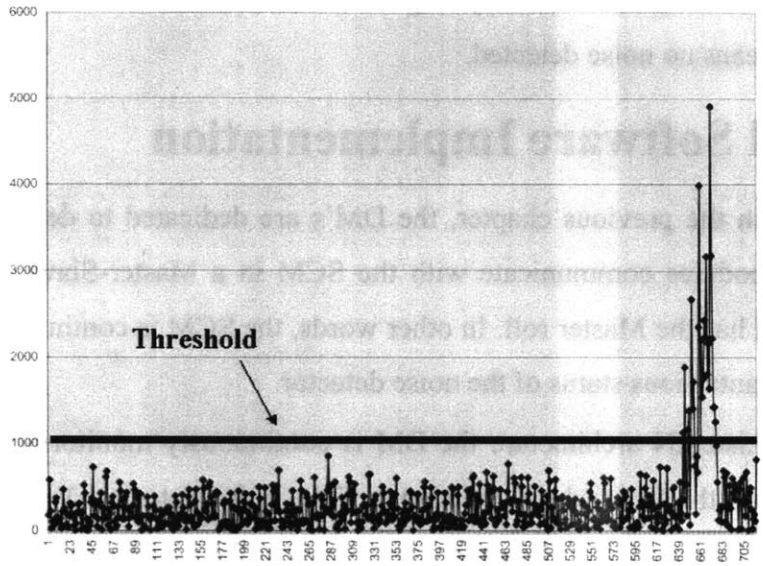

Figure 4-2 An example of Average Energy sensed by one Detection Module

Figure 4-2 shows an example output of the sensed Average Energy of localized audio.

Each point in the graph represents the average sound energy over a 32-millisecond window. When the average energy exceeds the threshold, the resulting burst denotes presence of new activity near to the DM. After a new energy burst is detected, the threshold is adjusted to twenty percent of its peak value. This dynamic adjusting of the threshold is done to avoid false detections.

4.2 Sound Collecting Module (SCM)

The SCM is implemented on a PDA. As previously discussed, this module uses the TB with the Nordic nRF2401 RF transceiver to communicate with the DM's and 802.11b

connectivity to send audio and data to the CM. The software is written in C/C++ using standard libraries.

4.2.1 SCM Hardware Implementation

The SCM is implemented on a PDA Compaq Ipaq H3600 running Linux. This module also uses the TB to communicate with the DM's. The TB is controlled by the PDA through its serial port. Figure 4-3 shows a picture of this module.

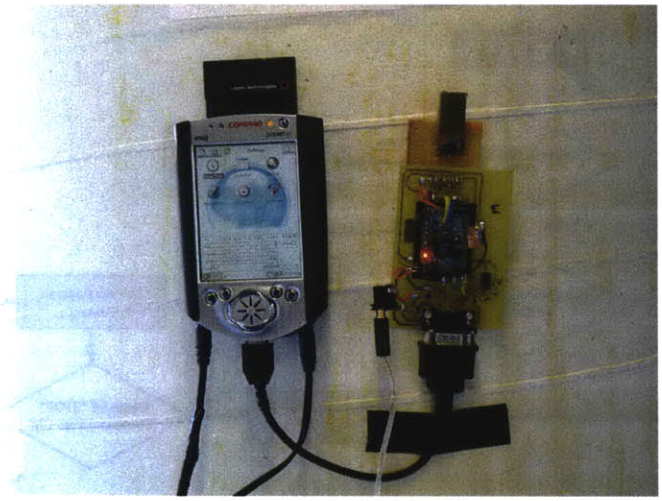

Figure 4-3 Sound Collecting Module (SCM): From left to right: PDA connected to TB

We use a PDA to develop this module because it offers the following advantages:

- Processing power. Adequate computing capabilities are needed to implement the algorithm to detect transitions in activity (changes in the average energy of the captured sound).
- 802.11b connectivity. Most PDA's have hardware designed for wireless communication.
- High quality sound device. The sound card of these devices often allows high quality recordings needed to experiment with the sound classifiers and Digital Signal Processing (DSP) techniques.
- Small form factor. The goal is to avoid interfering with inhabitant's daily activities at the monitored site.

The implementation of the TB is described in section 4.1.1

4.2.2 SCM Software Implementation

The software for the SCM is written in C\C++ using standard libraries. Here we explain the use and functioning of the three most important functions in the code.

Appendix A contains a link to the source code for this module. Figure 4-4 shows a flow chart of the software implemented in this module.

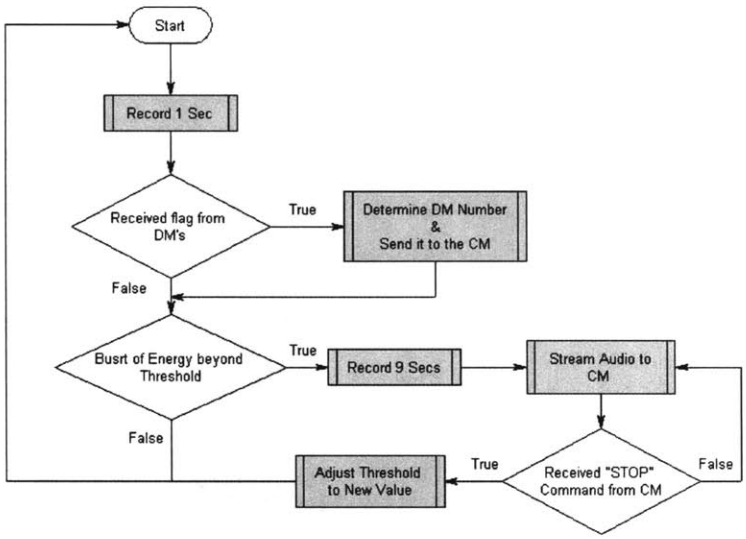

Figure 4-4 SCM software flow chart for the main program

The `main()` function is designed to stream audio to the CM whenever it detects a change in the energy of the room audio. This function is continuously recording one hundred-millisecond segments of raw audio. The recorded segments are temporally stored in an array under a Last Input First Output (LIFO) configuration. The temporal array stores 10 segments; this is one second of raw audio. After 10 segments are stored, the program starts left shifting the elements in the array. The `main` program continues capturing new segments until it detects a transition in the average energy of the noise then the program continues recording nine seconds of raw audio. The result is a piece of audio corresponding to one second before and nine seconds after the transition in activity is detected.

The function `send10secAudio()` is called when the `main` program detects a change in audio energy. This function sends the stored audio in the temporal array to the CM using 802.11b networking.

The function `StreamAudio()` starts streaming audio to be analyzed by the CM until the CM sends a command back to indicate the SCM that it should stop transmitting.

4.3 Classification Module (CM)

The CM is implemented on a Windows 2000 PC Pentium IV. This section focuses on the implementation of the key elements software wise that integrate the CM. The first subsection describes implementation of the AC. The next subsection describes the SD algorithm and its pseudo code. The last subsection describes how the entire software in the module was designed and implemented.

4.3.1 Activity Classifier (AC)

This subsection describes the implementation of the Activity Classifier. Activity Classification in ListenIN is done using three techniques: 1) Sound Classification using pattern recognition techniques, 2) Activity Classification based on: a) location of the sound source inside the house and b) sound duration and 3) Sound Classification based on detection of energy bursts and their temporal characteristics. For purposes of developing and implementing this system we built the "Taking a Shower" and "Brushing Teeth" classifiers using pattern recognition techniques; Cooking, Eating and Listening to Music classifiers use the technique number two; and a Baby Crying discriminator uses technique number three.

Sound Classification using pattern recognition techniques uses Nearest Feature Line (NFL) [30] and the well-known Nearest Neighbor (NN). To train these classifiers we recorded audio samples from activities such as taking a shower, and brushing teeth from real home environments and extracted thirty-four features as done in [30].

Activity Classification based on location of the sound source consists of collecting data from the Detection Modules (DM's). Since the CM keeps location of all DM's in a table, it associates DM's flags with activities. For example, if the CM detects that the DM located in the stove has a its flag set indicating presence of a sound source, then the activity is classified as cooking.

The third technique used in ListenIN is Sound Classification based on energy burst detection. This technique is implemented as described in [2].

4.3.2 Speech Detector (SD)

The SD is a speech/non speech discriminator, which essentially is looking for presence of vowels. Figure 4-5 shows the flow chart that illustrates the algorithm.

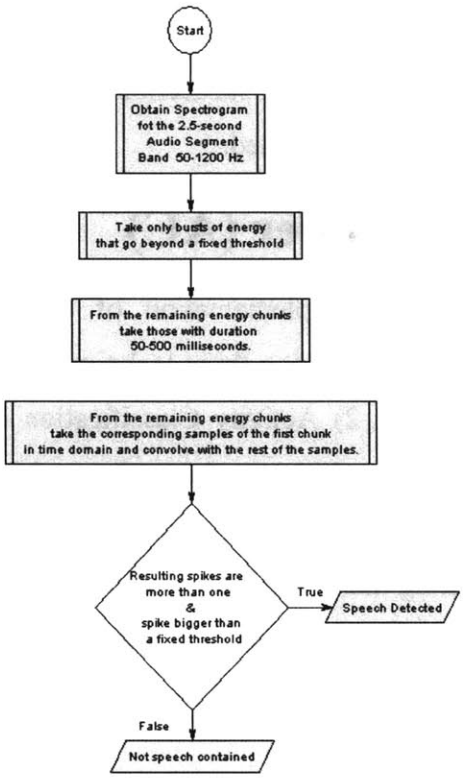

Figure 4-5 SD software flow chart

The algorithm first performs a frequency analysis on the original audio signal to detect bursts of energy of duration 50 to 500 milliseconds within the frequency band of 50– 1200 Hz. It then takes the samples in time domain corresponding to the first burst and convolves them with the remaining audio samples; the resulting peaks indicate similarity, which are taken as the presence of vowels.

4.3.3 General Software Description

This subsection describes the general design and implementation of the software that runs in the CM.

The CM utilizes the following classes: a) `CSndCollectMod` that implements functions to communicate with the SCM, b) `CAnalyzeSound` that uses instances of the Sound Classifiers and the Speech Detector to implement the Activity Classifier and the Speech Detector, c) `Cclient`, is the class that deals with the Client at the caregiver's side.

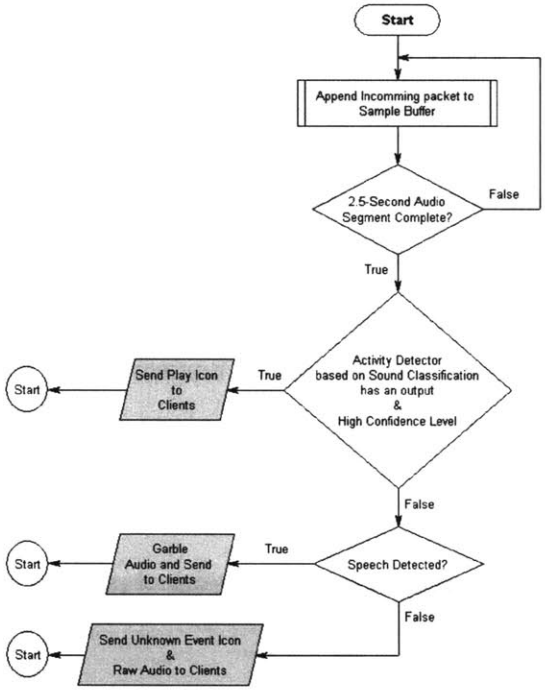

Figure 4-6CM software flow chart

Figure 4-6 shows the flow chart for the software implemented in the CM. The `CSndCollectMod` class when called increases the size of the TCP socket receiving buffer enough to store two seconds and a half of raw audio sampled in 16 bit mono format at 22050 Hz. This class is implemented using threads to indicate the compiler that this part and the rest of the code must be executed simultaneously. Basically this class is in charge of accepting connections and managing incoming packets from PDA's (SCM's).

The `CAnalyzeSound` class takes a 2.5-second audio segment and feeds it to the Activity Classifier (AC) and the Speech Detector (SD). This class evaluates the classifiers outputs and computes the confidence level. The confidence level is the minimal distance between the biggest output and the nearest smallest value. As shown in

the flow chart, the `CAnalyzeSound` class also decides what kind of information the CM should send to the subscribed clients.

Finally the `Cclient` manages the transfers of raw audio and audio icons id's to the subscribed clients.

4.4 Client Interface

The Client Interface developed for this thesis consists of a thin audio interface implemented on a PC and written in Visual Basic 6.0. This interface is the part of the ListenIN system that informs the caregiver about activity occurring at the server side.

The Client Interface receives commands and raw audio from the server using the TCP/IP protocol. This interface uses two socket connections: 1) to receive commands and 2) to receive raw audio from the server. All the alerts are locally cached and are played upon request from the server.

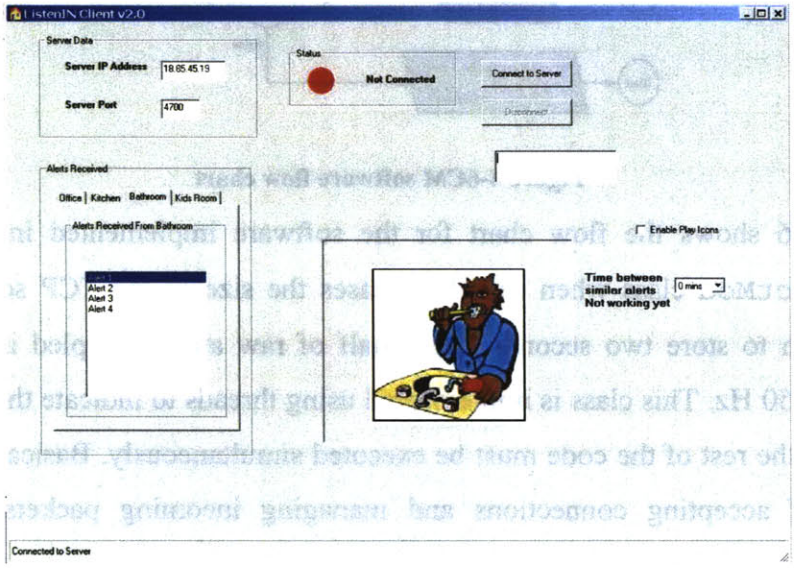

Figure 4-7 Screenshot of the Client Interface after receiving a Brushing teeth alert.

Figure 4-7 shows a screenshot of the Client Interface. Through the first socket, which connects to port 4700, the client receives commands such as "Play alert X" or "Play alert X and Receive raw audio of size Y". In the case of the second command, the client starts receiving audio on the second socket, which connects to port 4750. As soon as the client

interface receives all the data coming from the server, it plays the indicated alerts and/or raw audio.

Chapter 5

Evaluation

This chapter describes some evaluations on the user interface, the system architecture, and the system performance of ListenIN. The system went through several iterations of design, implementation and testing. The user that experienced the system was the author's academic advisor. He played the roll of caregiver receiving audio alerts through a Client Interface installed on his office PC. The rest of the system was deployed at his home. The evaluation here presented is a compilation of the lessons learned as the author's advisor informally tested the system. No formal user studies on this "proof of concept" system have been performed. In addition, this chapter also presents current limitations of the system and future improvements that could be made.

The first section of this chapter presents the tuning of parameters on the server. The second section describes the iterative process on the Client Interface to find the appropriate parameters that make using ListenIN a pleasant experience. Finally the last section describes the challenges faced to deploy ListenIN in a real home environment.

5.1 ListenIN Sever

This section describes the tuning of parameters on the server to increase the overall performance of the system.

5.1.1 Real Time Audio Processing

This subsection describes the iterative process to find the most efficient way to interface the server code with the detection algorithms. Inefficient performance of this section of the server may lead to congestion problems in the TCP receiving buffer.

Initially the server took approximately 9 seconds to process a 2.5-second segment of audio. Even though ListenIN was not originally designed as a real time interface, this delay caused major congestion problems in the TCP receiving buffer making the system unstable. After debugging the code of the `SndCollectMod` class, which is the class that takes care of the incoming audio streamed by the SCM's, we found that the main cause of the delay was that the sound classifiers were taking too much time to process audio compared to the original timing showed by the Matlab implementation.

In the beginning of the design process, we decided to use the Matlab C/C++ compiler to transform Matlab code into plain C code. This solution seemed to work well but later on when we tried to incorporate more than one SCM to the system we realized that such implementation was not thread safe, again, making the system unstable. Finally we decided to run Matlab in its mathematical engine mode and invoke it from the server code. We make these calls using a C/C++ API developed by MathWorks. The processing time was dramatically reduced to an average of 0.6 seconds. This solution is thread safe and faster than the original sound classifiers implementation in Matlab code.

5.1.2 Wireless link between DM's and SCM

This subsection describes the troubleshooting and debugging processes incorporated to make the wireless link between the DM's and the SCM work in a stable way. Wireless communication between the DM's and the SCM is extremely important to make the system portable and light weight.

As mentioned in previous chapters, the DM's communicate wirelessly with the SCM to transmit "detected/non-detected noise" flags that eventually arrive to the CM. At design time we choose the Nordic nRF2401 RF transceiver. This small transceiver (only 5 mm by 5 mm) presents many advantages: 1) Low cost, 2) Cyclic Redundant Check (CRC) hardware, a standard method for error checking and correction in data

communication and 3) Above all, 125 different channels to transmit. This feature avoids software implementation in the SCM to deal with multiple DM's talking at the same time.

During the implementation phase we encountered challenging tasks. Since this Integrated Circuit (IC) was new in the market at implementation time there was no source code available to aid the implementation of the control software. We therefore wrote the code from scratch, making debugging a painful task.

Another problem that we faced was the hardware implementation. Since even the transceiver's package is new (QFN24), a homemade PCB was required. Figure 5-1 shows the homemade PCB with the nRF2401 already soldered.

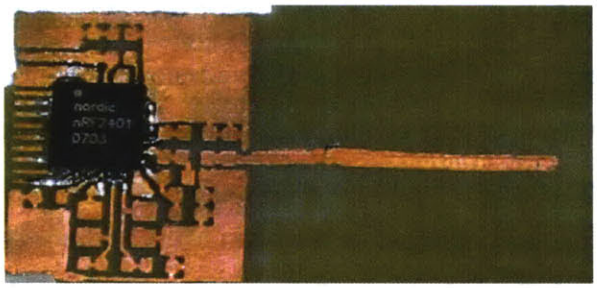

Figure 5-1 nRF2401 Homemade PCB construction

Due to its small size, soldering the components was time consuming. Figure 5-2 shows the final PCB with the transceiver mounted.

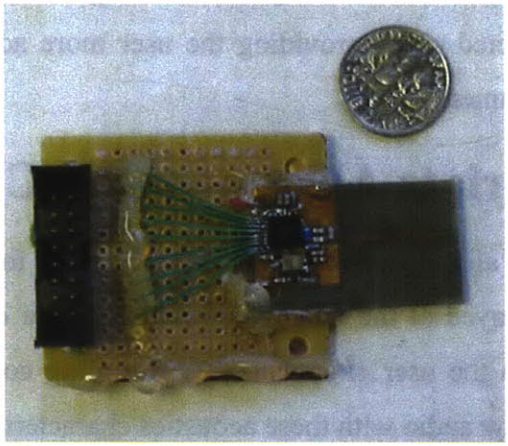

Figure 5-2 Final nRF2401 Homemade PCB

5.2 Client Interface

This section describes the iterative process the Client Interface went through after implementation. For purposes of evaluation, ListenIN was installed at the author's advisor home. The user experienced the system through his Client Interface installed at his office computer.

5.2.1 Appropriate Raw Audio Duration

Based on the obtained feedback we changed the original length of the raw audio that is sent to the client interface when the CM is unable to classify activity from the acoustic evidence that it obtained.

Originally this piece of audio consisted of five seconds before and five seconds after the transition in the average energy of the sound in the monitored room. The idea behind storing the recorded audio previous to the event that initiated the classification task is to give caregivers some acoustic context so they can classify the sounds that are receiving. After obtaining feedback from the user of the system we realized that the length of the displayed raw audio at the client side was too short: By the time the user noticed that a new event has being played the playback was near to end.

Based on this experience we changed the settings to one second before and nine seconds after the detected event providing the user more acoustic context information about the home environment.

5.2.2 Appropriate Alert Acoustic Characteristics

At the Client Interface, all the alerts and raw audio received from the server are wav files. Initially, all files displayed at the Client side maintained a constant gain through their duration. According to the user experience, the resulting combination of consequently displaying alerts and raw audio with these acoustics characteristics was unpleasant.

Having this in mind we decided to apply an attack and decay algorithm to the files before playing them. The attack and decay algorithm consists of applying an envelope to the original audio signal that grows rapidly at the beginning, then becomes flat, and finally decays in a subtle way. An example of the resulting signal is shown in figure 5-3.

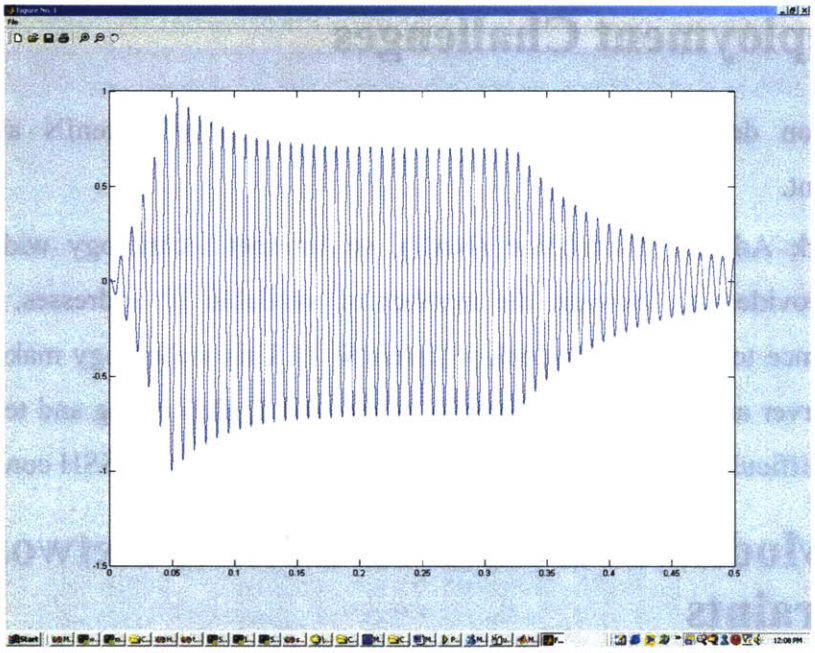

Figure 5-3 Example of an Audio signal with Attack and Decay envelope

5.2.3 Timing Between a particular kind of Alert

Another important parameter of the Client Interface is the time that the interface has to wait before displaying similar alerts.

As mentioned in previous chapters the processing task at the server side is initiated each time a new transition in the average energy of the sound room is detected. This design feature has an undesired consequence at the Client side: In a room where inhabitants are very active, so much audio is produced that the server is almost continuously processing audio and generating outputs that are eventually sent to the Client Interface. The user reported that this continuous playing of sounds was confusing and somewhat unpleasant.

This obviously is a design error and to solve it we decided to attack the root problem. We modified the original algorithm that runs in the SCM. Originally this algorithm continuously looked for changes in activity so we decided to insert a 60 second delay after the detection process finished. This simple solution prevented the server from sending continuously the same kind of alert in a short amount of time, relieving users from an annoying experience.

5.3 Deployment Challenges

This section describes the challenges faced to deploy ListenIN at a real home environment.

Network Address Translation (NAT), an Internet technology widely utilized by Internet providers to increase the number of available IP addresses, was the major inconvenience to successfully deploy ListenIN: 1) This technology makes it difficult to set up a server at the user side of the Network, and 2) Debugging and testing new SCM code was difficult since NAT kept PDA's unreachable to remote SSH connections.

5.3.1 Modified Architecture due to Network Constraints

The architecture of ListenIN was modified from its original design to be deployed at a real home environment. Figure 5-4 shows the architecture finally deployed and tested.

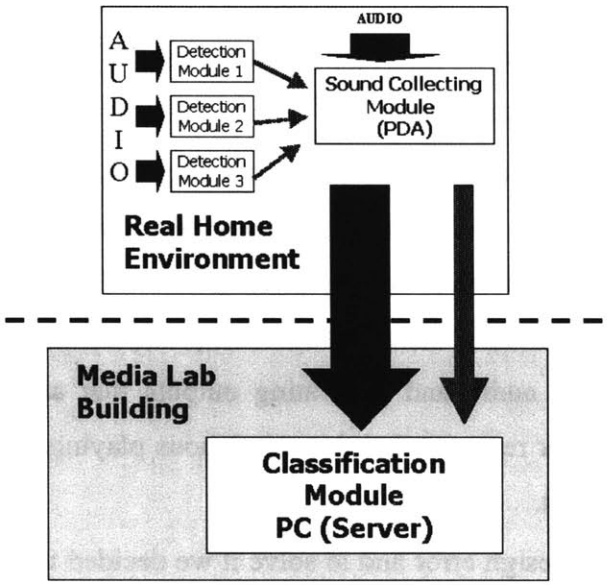

Figure 5-4 Modified ListenIN Architecture deployed at a real home environment for evaluation

Due to the network constraints mentioned above, the server, which was supposed to live at the monitored home, was moved to inside the Media Lab Network.

The PDA remained at the monitored home streaming audio and data using 802.11b networking to a wireless router; then the packets were sent through the Internet to the server.

5.3.2 Constraints implicit to Network Address Translation (NAT)

The major challenge was to deal with the implicit constraints to the use of NAT. This Internet technology does not allow easily setting up a server due to its functionality mainly focused on assigning IP addresses on network sites where most of the applications are clients such as Internet browsers and conferencing tools.

This constraint forced a major change in the original architecture of ListenIN: The server, which was supposed to live at the monitored home, was moved to inside the Media Lab Network. This change brought to light a problem in the `SndCollectMod` class. Originally, the receiving buffer in this class was flushed after processing audio. Part of this problem was due to different network bandwidth characteristics in the Media Lab vs. home environment. Although well below the WiFi data rates, audio from the PDA saturated the local router or the upstream cable modem connection at home. This bottleneck, not seen in the Lab, caused unwanted behavior in the PDA software, which had to be debugged. This shows some of the difficulty of deploying lab systems into a real home environment. In summary, under the Media Lab network the system performed well, but under the modified architecture flushing the acquired data caused the server to crash. Debugging and finding the root problem was a difficult task. To solve the problem, we simply removed the flushing function from the code.

Another problem derived from the use of NAT by the Internet provider is that the PDA was unreachable for SSH connections from the Media Lab, where all the development tools are installed, making debugging and testing new code for PDA's a challenging task. To overcome this inconvenience we wrote more code for PDA's to constantly ask the server if there was a new version of the SCM software. We also wrote code for a remote downloader in the SCM's.

Chapter 6

Conclusion

This chapter closes the work presented in this thesis. Here we present contributions and future work that may be done to improve performance of ListenIN.

In summary, in this thesis we proposed a system that allows closely related people, such as relatives and friends, to maintain awareness of a distant site such as home through auditory cues and eavesdropping. Through Internet connections activity information can be presented as an audio icon, a pre-recorded piece of sound representing a particular activity (e.g. a recording of a baby crying), instead of the actual sound, on a desktop or any mobile device having an Internet connection.

The first section of this chapter describes the contributions of ListenIN. The second section is a compilation of the future work that may de done to improve performance and user interface of ListenIN.

6.1 Contributions

Existing systems that attempt to maintain awareness at remote places require complicated setups at the monitored side such as installing multiple cameras, wiring sensors or even worse they require that subjects under observation wear sensors. Such systems are not portable and they interfere is inhabitant's daily life.

ListenIN is an effort to overcome the complexity that usually keeps a system from being portable. There is obviously a trade-off between the complexity of setting up a

system and the amount of information it can capture and process. ListenIN hypothesizes that enough information to classify activity can be extracted from audio signals alone. Moreover, audio by itself is an attractive medium due to its low processing complexity as compared to other media.

Little work has been done using audio to represent recognized activity. ListenIN implements this idea by delivering, a pre-recorded piece of sound representing a particular activity (e.g. a recording of a baby crying), instead of the actual sound. At the caregiver's side, this audio icon conveys enough information to associate it with a particular activity while maintaining privacy at the monitored side.

Finally, another beneficial outcome of this research is a robust algorithm to discriminate speech from non-speech sound in noisy environments. The ListenIN architecture requires such a discriminator. Originally we planned to use common Digital Signal Processing techniques to implement it. However, we realize that existing techniques were designed assuming a Signal to Noise Ratio (SNR) that by far exceeds real conditions in a home environment.

6.2 Future Work.

The following is a list of future work that would make ListenIN a more efficient and exciting tool to taker care of remote relatives.

Server Hardware

- The size of the Transmitting Board (TB) used by the DM's and SCM's could be reduced. For purposes of fast prototyping the TB presented in this thesis was done using a homemade etching technique that lacks of resolution to miniaturize the TB.
- The size of the DM's could be dramatically reduced if redesigned to build a new PCB. The hoarder board offers advantages such as fast prototyping but for purposes of implementing DM it is an over kill.

- Including a primitive input device in the DM's such as a push button could make it possible to easily assign and change the identification number to each DM. At this time id numbers are programmed at compilation time.

Server Software

- Several classification techniques could be evaluated to build new sound classifiers and/or replace the existing ones. Techniques such as Support Vector Machines, Hidden Markov Models or variants to these techniques could be explored under the promise that better recognition results can be achieved.
- Sound Classifiers and Speech Detector algorithms can be directly implemented in C/C++ to improve performance and processing speed.
- Audio encoding would be beneficial especially for varying network connectivity where high bandwidth is not always available.

Client Interface

- Choosing appropriate alerting is difficult especially with increasing number of activities that can be recognized. However, Mynatt's research [8] indicates that users want to choose their own alerts. Allowing users to choose they own alerts to represent activities could enhance the Client Interface.
- Including a history option in the Client Interface would help caregivers to keep track of important events related to their relative's health.
- The architecture of ListenIN allows that almost any device with an IP address can subscribe to the system. A project developed by Natalia Marmasse will use a Motorola Iden phone as a mobile client of ListenIN. Since Iden phones have some constraints such as storage capacity and control over the sound device, a mini sound card for Motorola Iden phones was developed at the Speech Interfaces Group.

We want to remark that ListenIN does not contribute to solving the problem of accurate activity recognition using pattern recognition techniques. ListenIN can be considered as a new attempt to maintain significant awareness for caregivers to infer that

the person being monitored is generally engaged in normal activity. The novel architecture of ListenIN allows for an easy integration of different kinds of sound classifiers and has no intention of replacing social communication; it is intended to be a tool to help caregivers know dependent's urgency of human contact or assistance.

Appendix A

Source Code

This Appendix contains links to web pages were the source code for the Sound Collecting Module (SCM), Detection Module (DM), Transmitting Board (TB) and the server can be accessed.

- The source code for the Sound Collecting Module (SCM) can be downloaded from http://web.media.mit.edu/~gvallejo/listenin/scm.htm
- The source code for the Detection Module (DM) is available for download at http://web.media.mit.edu/~gvallejo/listenin/dm.htm
- The source code for the Transmitting Board (TB) can be downloaded from http://web.media.mit.edu/~gvallejo/listenin/tb.htm
- The source code for the ListenIN server can be found at http://web.media.mit.edu/~gvallejo/listenin/server.htm

Appendix B

Audio Formats

ListenIN uses 22050 Hz, 16 bit, mono for audio processing on PDA's, the Server and the Client. The Server sends and receives audio in RAW format. The Client receives RAW format and converts it into WAV format without changing sampling characteristics.

The Client uses `sox` to convert the received audio from RAW format into WAV. The command to do the conversion is

```
sox -V -r 22050 -s -w -c 1 inputfilename.raw outputfilename.wav
```

The Detection Module (DM) uses 8 Khz, 8 bit, mono to process audio and perform noise detection.

References

[1] C. Schmandt, G. Vallejo. "ListenIN" to Domestic Environments From Remote Locations, *Proceedings of the 9th International Conference on Auditory Display,* 2003: 220-223.

[2] C. Schmandt, J. Kim K. Lee G. Vallejo M. Ackerman. Mediated Voice Communication via Mobile IP, *UIST International Conference,* 2001: 141-150.

[3] D. Hindus and C. Schmandt. Ubiquitous Audio: Capturing Spontaneous Collaboration, *Proceedings of ACM CSCW'92 Conference on Computer-Supported Cooperative Work,* 1992.

[4] D.J. Moore, I. A. Essa and M. H. Hayes III. Exploiting human actions and object context for recognition tasks, *Proceedings of the 7th IEEE International Conference on Computer Vision,* 1999.

[5] D. R. Rutter. The role of cuelessness in social interaction: An examination of teaching by telephone, Conversation: An Interdisciplinary Perspective. 1989.

[6] E-Care. E-Care Assitant., http://www.livingindependently.com

[7] E. D. Mynatt, J Rown A. Jacobs and S. Craighill. Digital Family Portraits: Supporting Peace of Mind for Extended Family Members, *Proceedings of ACM CHI'92 Conference on Human Factors in Computing Systems,* 1992: 541-547.

[8] E. D. Mynatt, M. Back R. Want M. Baer and J. B. Ellis. Designing Audio Aura, *Proceedings of ACM CHI, Los Angeles CA,* 1998: 566-573.

[9] E. Munguia. Activity Recognition in the Home Setting Using Simple and Ubiquitous Sensors, *M.S. Thesis, Massachusetts Institute of Technology, 2003.*

[10] F. Foerster and J. Fahrenberg. Motion pattern and posture: correctly assessed by calibrated accelerometers. *Behavior Research Methods, Instruments, & Computers,* 2000: 32(3)450-7.

[11] I. Haritaoglu, D. Harwood and L. Davids. (w)4: (w)ho, (w)hen, (w)here, (w)hat: (a) real time system for detecting and tracking people, *Third International Conference on Automatic Face and Gesture,* 1998.

[12] I.Smith, S. Hudson. Low Disturbance Audio for Awareness and Privacy in Media Space Applications, *Proceedings of ACM Multimedia'95,* 1995.

[13] ICQ. ("I seek you"). http://www.icq.com

[14] J.C. Tang, E. A. Isaacs and M. Rua. Supporting distributed groups with a Montage of lightweight interactions, *Proceedings of CSCW,* 1994: 23-34.

[15] J. Davis and A.F. Bobick. The representation and recognition of action using temporal templates, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* 1997: 928-934.

[16] J. H. Tulen, D. L. Stronls J. B. Bussmann L. Pepplinkhuizen and J. Passchier. Towards an objective quantitative assesment of daily functioning in migraine: A feasibility study, *Pain,* 2000: 86(1-2):139-149.

[17] J. Watlington. Synthetic Movies, *MS Thesis, Massachusetts Institute of Technology,* 1989.

[18] K. Nagel and G. ABowd. The Family Intercom: Developing a Context-Aware Audio Communication System, *Proceedings of Ubicomp 2001 International Conference,* 2001.

[19] M. Ackerman, D. Hindus S. Mainwaring and B. Starr. Hanging on the 'Wire: A field study of an audio-only media space, *ACM Transactions on Computer-Human Interaction,* 1997: 39-66.

[20] M. Makikawa, S. Kurata Y. Higa Y. Araki and R. Tokue. Ambulatory monitoring of behavior in daily life by accelerometers set at both-near-sides of the joint, *Proceedings of Medinfo,* 2001: 840-843.

[21] MathWorks. MatLab, the language of technical computing. http://www.mathworks.com

[22] Nordic. The Nordic VLSI nRF2401 ultra low power 2.4GHz transceiver, http://www.nvlsi.no, 2003.

[23] P. Dourish, S. Bly. Portholes: Supporting Awareness in a Distributed Work Group, *Proceedings of ACM CHI'92 Conference on Human Factors in Computing Systems,* 1992: 541-547.

[24] R. Beckwith and S. Lederer. Designing for one's dotage: Ubicomp and residential care facilities, *Home Oriented Informatics and Telematics (HOIT),* 2003.

[25] S.Hudson , I. Smith. Techniques for addressing fundamental privacy and disruption tradeoffs in awareness support systems, *Proceedings of the ACM conference on Computer Supported Cooperative Work,* 1996: 248-257.

[26] S.S. Intille, J. Davis and A. Bobick. Real-time closed world tracking, *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition,* 1997: 697-703.

[27] S. Stillman, R. Tanawongsuwan and I. Essa. A system for tracking and recognizing multiple people with multiple cameras, *Proceedings of the Second International Conference on Audio-Vision-based Person Authentication,* 1999.

[28] S. Whittaker, D. Frohlich and O. Daly-Jones. Informal Workplace Communication: What Is It Like and How Might We Support It?, *Proceedings of ACM Conference on Human Factors in Computing Systems CHI'93,* 1994: 131-137.

[29] S. Xinding and C. Ching-Wei. Probabilistic motion parameter models for human activity recognition.

[30] S.Z. Li. Content -based classification and retrieval of audio using the nearest feature line method, *IEEE Transactions*, 2000: 619-625.

[31] T. Choudhury and A. Pentland. The sociometer: A wearable device for understanding human networks, *Media Lab tr 554, Massachusetts Institute of Technology,* 2002.

[32] V. Gerasimov. Hoarder aka Swiss Army Knife (SAK) board. 2001. http://vadim.www.media.mit.edu/Hoarder/Hoarder.htm

[33] W. Gaver, R. Smith T. O Shea. Effective Sounds in Complex Systems: The ARKola Simulation, *Proceedings of ACM Conference on Computer Human Interaction,* 1991: 85-90.