Monkey Business:

Creating social awareness among distributed group members, using a network of animatronic agents

by

Rachel Lori Kern

Bachelor of Arts in Cognitive Science
Northwestern University, 1999

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 in Media Arts and Sciences

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2006

Author.....
Program in Media Arts and Sciences,
May 22, 2006

Certified by.....
Christopher Schmandt
Principal Research Scientist
M.I.T. Media Laboratory
Thesis Supervisor

Accepted by.....
Andrew Lippman
Chairperson
Department Committee on Graduate Students
Program in Media Arts and Sciences

Monkey Business:

Creating social awareness among distributed group members, using a network of animatronic agents

by

Rachel Lori Kern

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning
on May 22, 2006, in partial fulfillment of the
requirements for the degree of
Master of Science in Media Arts and Sciences

Abstract

Members of a geographically distributed group are not normally aware of each other's presence or current activities. For example, two members of a team may be working on the same project, but they may have offices in different parts of a building. This geographical separation prevents them from knowing when the other has arrived in the morning, or if the other is busy or available, and it generally leads to a lack of awareness about the other's activities. It also tends to limit spontaneous and informal interaction among teammates.

For this thesis, I have built a prototype of a system to keep distributed members of a group aware of each other's presence and activities in a light-hearted manner, while striving to remain non-intrusive. The system also aims to facilitate unplanned and informal communication among distributed colleagues. It consists of a network of animatronic agents, specifically monkeys, which are situated in the offices or rooms of each member of a group. Through subtle movements and sounds, the monkeys indicate the presence of the other members of the group. The monkeys are meant to be ambient, at the periphery of one's attention. But they can also be used more proactively as communication mechanisms, and promote informal exchanges among members of a distributed team.

The objective of this research is to consider whether such a system can be helpful in keeping members of groups more connected and in providing greater social awareness and cohesiveness among them. I have also explored whether animatronic agents are a good medium for communicating useful ambient information in a non-disruptive manner, and if they are capable of facilitating spontaneous communication. Finally, I have tried to determine the right combination of motion and sound in order for the monkeys to communicate information effectively and intuitively among group members.

Thesis Supervisor: Christopher Schmandt
Title: Principal Research Scientist, M.I.T. Media Laboratory

Thesis Committee

Thesis Advisor: _____

Christopher M. Schmandt
Principal Research Scientist
M.I.T. Media Laboratory

Thesis Reader: _____

Hiroshi Ishii
Associate Professor of Media Arts and Sciences
M.I.T. Media Laboratory

Thesis Reader: _____

Joseph Dvorak
Research Affiliate
Motorola Scientist in Residence

Acknowledgments

As I sit here at my desk at the Media Lab, watching an animatronic monkey in my office looking up at me, shrieking in monkey tones, and waving its arms around, I can't help but think back to about a year ago, when my advisor first suggested that I incorporate animatronics into my thesis project. Though I thought his suggestion sounded intriguing, admittedly my initial reaction also included some doubt and uncertainty. I did not have much experience working with hardware. I didn't know the first thing about servo motors or micro-controllers. Even assuming I could learn everything I needed to know to get the project to work (which is one test of a good Media Lab student), I wondered if anyone would take an animatronics project seriously. I had seen positive reactions to the Cellular Squirrel (the predecessor of the monkeys), and I agreed that it was cute, but I still didn't really "get" it. I had many questions: Could an animatronic agent actually be useful? Could it serve an important purpose, other than simply being cute and making people smile? Wouldn't people regard it as just a silly toy?

Now that my Monkey Business Masters thesis is nearing completion, I realize that when I first started thinking about the project a year ago, I was largely missing the point. The *point* is that the monkey agents are cute and silly and make people smile. They have a "practical" purpose as well – they are meant to keep distributed group members aware of each other's presence, and they are designed to promote spontaneous and lightweight communication between geographically separated parties. But what better way to promote lightweight communication than by making people smile and laugh?

I owe a huge debt of gratitude to the many people who were instrumental in helping to make Monkey Business a reality:

I would like to thank my advisor, Chris Schmandt, for admitting me to the Media Lab, even though it took him two tries to create a spot for me in his group. I have learned a tremendous amount from Chris on a wide variety of topics, including the history of area codes in Massachusetts and the intricacies of patent lawsuits, as well as a thing or two about voice communications. Chris has been a great advisor, always interesting to talk to, never afraid to say what he really thinks, and with a great sense of humor to boot. It has been a real joy to work in his group.

I would also like to thank my other thesis readers, Hiroshi Ishii and Joe Dvorak, for their time and insights into my work. Hiroshi's Tangible Interfaces class helped to show me the value in having an ambient, physically embodied interface, and convince me that a three-dimensional monkey could be much more effective than a two-dimensional monkey on a screen. Joe's energetic enthusiasm for my work helped to augment my own enthusiasm for it. He provided me with a great opportunity to demonstrate my project to other Motorola employees, and he was always helpful in coming up with new ideas and suggestions for how the animatronics could be used.

I am extremely thankful to everyone who helped out with the Monkey Business project – it was truly a team effort. First, thanks to Stefan Marti, for building the skeletons of the monkeys, teaching me to solder, to build audio amplifiers, to program servo controllers, and pretty much everything else I needed to know about hardware and animatronics. His work on the Cellular Squirrel served as the inspiration and foundation of my work, and I am extremely grateful for his tireless willingness to help, even from the other side of the country.

Thanks also to Paulina Modlitba, who is writing a Monkey Business thesis of her own. Paulina helped in building the monkeys and the project webpage, and most importantly, in designing and running the user studies. She also helped to keep me on track, and to have a sense of humor about this project when it started to overwhelm me. Finally, she provided me with female companionship in a largely male environment, and was a great addition to our group, encouraging us all to get to know each other better outside of the lab.

Thanks to Jae-woo Chung, who has been my officemate for the past two years. Last year, Jae-woo and I made a deal – he would help me with Java, and I would help him with English. Jae-woo has helped me with much more than Java. Here are just a few of the things he has done: edited videos of my project, taught me to use PhotoShop for making posters, edited monkey sounds to make the monkeys more expressive, and given me useful feedback and suggestions for all of my projects along the way.

Chaochi Chang was my other officemate during my second year here at the lab. Chaochi has played a number of integral roles in the Monkey Business project. He figured out how to incorporate motion and proximity sensors into the system, wrote most

of the code for the SuperMonkey Server, and also wrote code to analyze audio to be played over the system. His patient manner has made him very easy to work with. Chaochi has also been a reliable source of Red Sox trivia throughout the year.

Matt Hofmann, now a senior at MIT, has been a dedicated UROP in the Speech Interfaces group since he was a freshman. Matt wrote the streaming audio code for Monkey Business. In addition to starring in many of our group's videos, Matt is great at explaining things and making complicated concepts easy to understand.

Finally, I would like to thank the two most recent additions to our group, Matt Adcock and Jeff Goldenson. They will likely inherit the monkeys after I graduate. Matt took an interest in the project from the moment he got here, and has been a constant source of feedback, ideas, and interesting links. In addition to being very kind to help out with my job search, Matt also introduced the group to the famous Australian Tim Tam Slam. Jeff graciously allowed me to put a monkey in his office during his second week with the group, and fortunately, he seemed to enjoy having it there.

I am also thankful to all of the other students, faculty, and staff who helped me during my time here – whether it was by participating in a user study, lending me a book, a power supply, or good advice, coming to hear my lunchtime talk, or just saying hi in the hallway, you have made my time at the Media Lab more fulfilling and enjoyable!

Thanks to all of my friends for encouraging me, and always having confidence that I would succeed, even when I was doubtful. Thanks to my brother Josh, who helped me with EMotoPhone, my first project at the lab. Josh definitely has the talent and creativity to be a Media Lab student himself, but since he doesn't like to follow in my footsteps, we'll see if this ever happens. Lastly, a huge thanks to my parents for everything they have done for me: patiently proofreading all my papers (including this one), providing me with meals when I was too stressed out to cook, and always being my biggest fans and supporters. It was my Mom who encouraged me to apply to the Media Lab in the first place, despite my conviction that there was no way I would ever get in; I'm very grateful that she persuaded me to try anyway. I'm even more grateful that I proved myself wrong, and once again, I am extremely thankful to all of you who have made my time here so inspiring.

Table of Contents

List of Figures and Tables

Chapter 1

Introduction

1.1 The Problem

Imagine the following fictional scenario: Larry and Kate are colleagues who share an office. Their supervisor, Mark, has an office right next door. All three are working together on a project along with a fourth team member, Susan, whose office is on the other side of the building. Once a week, the team has pre-arranged meetings to update each other on the status of their particular components of the project. Because of the proximity of their offices, Mark also often stops by Larry and Kate's office to discuss the progress of the project and to inform them of any new developments. Additionally, Larry and Kate frequently break into spontaneous discussions about their work. Sometimes they invite Susan over to their office when one of these conversations escalates into a more important dialogue. But more often than not, Susan is left out of these informal, spur-of-the-moment discussions. The group does not exclude her intentionally; rather the exclusion occurs naturally as a result of Susan's remote office location in a different part of the building. Short of rearranging the entire office space to better accommodate every distributed workgroup in the whole building, is there a way to include Susan in these impromptu communications? How can Susan be made aware that an unplanned, yet significant interaction is taking place among her colleagues, without them telling her about it outright?

Many studies have shown that informal communication resulting from physical proximity serves many purposes, and accordingly yields many benefits. While some of these purposes are work-related (i.e. coordinating meeting times, updating colleagues about new developments in a project), informal communication is also often used as a vehicle for building camaraderie, forming social bonds, and developing cohesiveness among groups. Working in the same physical environment helps foster a sense of community and connectedness among colleagues, and this helps to keep members of a workgroup happy and committed to the projects on which they are working [13]. Co-located people often benefit from chance encounters and spontaneous discussions, which

enable them to discover shared interests, and to exchange information in an informal and non-intrusive manner [1, 6]. Furthermore, greater proximity among colleagues leads to a greater familiarity with each other's work, and consequently, a greater respect for each other's work [13].

Informal communication and physical proximity also enable colleagues to acquire greater knowledge of the state of each other's work, which helps in preventing miscommunications and avoiding potential problems [17]. Working in the same physical space can potentially minimize the need for interruptions; for example, there is no need to ask what somebody else is doing at the moment, as this information can be gleaned by a quick glance. Though informal communication is inherently brief and unplanned, it is also very frequent. In fact, in a study of workplace communication, it was found that informal communication accounts for as much as 31% of work time [22].

Unfortunately however, the reality of the modern-day workplace is that many colleagues are not co-located, and thus, they have little or no opportunity for face-to-face informal communication and lose out on its resultant benefits. With the advent of telecommuting, and global companies with offices spread all over the world, workers are often separated both in space and in time from their fellow workers. Even within the same office building, as the fictional scenario above depicts, colleagues' offices may be in different parts of a building, or on different floors. Despite being a shorter distance, this division can still be a hindrance to reaping the benefits of unplanned, informal and spontaneous communication that results from proximity.

Physically distributed colleagues must exert extra effort to stay aware of their teammates' progress and activities. Communication necessarily becomes more planned, and as a result, more formal. Geographical separation tends to limit the spontaneous and informal exchange of ideas, and as a result, undermines the overall cohesiveness and effectiveness of a workgroup [17].

This problem is not limited to workplace communications. Personal relationships with friends, significant others, and family also benefit from proximity and the ability to make spontaneous, last-minute plans. These relationships may suffer when large distances are introduced among the parties. Instead of enjoying brief, frequent, and informal communications, the parties must plan time to talk when neither is busy, and

must often set aside large chunks of time to catch up, especially when a great deal of time has elapsed since their last conversation.

1.2 Proposed Solution

This thesis describes a system that attempts to keep distributed members of groups more closely connected and aware of each other's activities in an informal and light-hearted manner. The system aims to facilitate informal and spontaneous communication among group members who are geographically separated. It attempts to do so while also minimizing interruption at inopportune times. The goal of the system is to enable distributed group members to reap the benefits of the informal communication that occurs naturally as a result of physical proximity, even when it is not possible for every member of the group to be physically proximate. It is important to note that many other systems have also been built with similar goals in mind, using a variety of different techniques of keeping distributed group members connected. I will highlight some of these systems in Chapter 2.

Our system approaches this problem from a rather unique angle – it consists of a network of animatronic agents, such that one agent resides in the office of each member of a distributed group. I have chosen the embodiment of a monkey for the form of these agents, hence Monkey Business as the title of this work. The monkey agent uses a combination of microphones and sensors to recognize if activity is occurring in the office that it occupies. If there is a change in state of the office activity, the monkey transmits this information out to the network of other monkeys. The other monkeys, through subtle gestures, movements, and sounds, indicate the change of state in the transmitting office. Thus all members of the distributed group, through their respective monkeys, are made aware of each other's activities and presence in an ambient and light-hearted manner.

If the monkey in one office makes its owner aware of a particularly interesting conversation, or if there is a significant change of state in another office, the owner may wish to learn more about what has occurred in the other office. In this case, he may actually use the monkey's audio capabilities as an intercom, and communicate to members of his group in other offices via his monkey. He indicates that he wishes to speak to his teammates by leaning toward his monkey; the monkey senses this leaning-in

motion with a proximity sensor, and subsequently opens an audio channel to the other monkeys in the network. The user can then speak, and his voice will be broadcast through the speakers in all the other monkeys.

The goal of the Monkey Business system is for the light-hearted and playful character of the monkeys to encourage distributed groups of people to use them for the kind of spontaneous and informal communication that does not usually take place when people are not physically proximate.

The remainder of this thesis is organized as follows: In Chapter 2, I will discuss the inspirations for the Monkey Business System, as well as some of the related research projects that have been carried out in a similar vein. Chapter 3 gives a more in-depth system description, along with the technical details of the architecture of the system, and the integration of all of the system components. Chapter 4 describes several hypothetical scenarios of how the Monkey Business system might be used in different settings and circumstances. Chapter 5 presents the user studies we undertook, and analyzes the results and feedback we obtained from these studies. Finally, in Chapter 6, I draw conclusions that I have made from this research, and also highlight areas for future work to be done to further build upon and improve the Monkey Business system.

Chapter 2

Related Work

2.1 Inspirations

Monkey Business was inspired mainly by three previous research projects, which were also carried out in the Speech Interface Group at the M.I.T. Media Lab, namely ListenIN, SimPhony, and the Cellular Squirrel. Here I will briefly describe these systems, and how each of them contributed to the Monkey Business project.

ListenIN is a system that uses auditory cues and eavesdropping techniques to allow people to listen to bits and pieces of the activity happening at distant locations. It is primarily intended for use by close friends and family members, since it can give a very intimate view into another's life. One practical application of the system is to allow caregivers to monitor the activity of their dependents remotely, and to be able to tell immediately if a dependent has any urgent needs. In the ListenIN system, the monitored location (which is most often a home) is equipped with strategically placed wireless microphones. The microphones monitor the activity in the home, and whenever there is a change in the current activity (such as watching TV, cooking, or leaving the house), the system sends a brief audio clip to the listener, or caregiver, thus making him or her aware of what is happening at the remote site [21].

SimPhony is a mobile voice communication system, specifically intended for distributed workgroups. It is similar to an instant messaging client, though it uses voice as a medium of communication instead of text. Users can access the system via either PDAs or through telephones. They can send voice instant messages either to individuals (buddies), or to pre-specified groups of buddies, such that everyone in the group receives the message. When there is a frequent exchange of asynchronous voice messages between two parties, the SimPhony system automatically transitions to a synchronous mode, so that the parties will start having an audio chat in real time. Users also receive audio notification whenever a new buddy logs into the system, or whenever one of their pre-specified buddy groups becomes active [14].

The Cellular Squirrel, also known as an "autonomous interactive intermediary," is a telephone agent, embodied as a small, wireless, animatronic squirrel, as shown in Figure 2-1. The squirrel handles a user's telephone calls as they come in, and also alerts a user to new calls; the squirrel's actions differ depending on whether the user is available to take a phone call the moment it comes in, or whether he or she is currently occupied by a conversation with a co-located individual. The squirrel's default behavior is to sleep, gently moving its head up and down. When a phone call comes in, the squirrel wakes up by lifting its head and looking around, almost as if to try to make eye contact with the user. This subtle, human-like method of getting the user's attention was found in a user study to be a less jarring and intrusive interruption than a ringing phone, both to the user, and to any co-located individuals who may have been talking to the user when the call came in. However, unlike a vibrating phone alert, which is perceptible only to the user, the squirrel's notification is public, so that co-located parties are also made aware of the incoming call. This makes for a less awkward transition if the user elects to take the call, thereby ending his or her conversation with any co-located people [19].

Figure 2-1. The Cellular Squirrel

Monkey Business draws on components from all three of these projects. From ListenIN, it borrows the idea of listening to activity in remote locations, and giving updates when changes in state are detected. Like ListenIN, Monkey Business also uses microphones to detect activity occurring at remote sites. Monkey Business then sends actual recorded audio from one location to another, which is also similar to the operation

of ListenIN. But unlike ListenIN, Monkey Business uses sensors for detecting changes in state, in addition to audio. Monkey Business also broadcasts changes in state using an agent's movements and gestures, as well as sounds, whereas ListenIN relies solely on sounds for this purpose.

The intercom capabilities of the monkey agents in the Monkey Business system are reminiscent of the voice group communication that SimPhony provides. Like SimPhony, people can use the monkeys for both one-to-one chats, and one-to-many chats, in which one person addresses a group. Unlike SimPhony, Monkey Business is not a mobile system, and it only supports synchronous communication; there is no option for leaving asynchronous messages.

Finally, the idea of using animatronic monkeys as agents in this project was based upon the success of the animatronic cellular squirrel as a telephone agent. Though the monkey and the squirrel differ in function, they share the property of using an appealing physical embodiment as an awareness tool, to convey information to humans in a subtle and ambient manner.

The ideas explored in the ListenIN, SimPhony and Cellular Squirrel projects provide a solid foundation for Monkey Business to build upon. These three projects have been instrumental in inspiring and motivating the work described here.

2.2 Systems With Similar Goals

Other research groups have worked on projects with goals similar to that of Monkey Business: promoting background awareness and informal interaction among distributed group members. These projects have explored a variety of different media in trying to achieve this goal, with varying degrees of success.

Thunderwire, a project done at Interval Research in collaboration with the University of California at Irvine, experimented with using an audio-only system to support informal interaction and collaboration among members of a workgroup. The system had no visual interface, other than an on/off light, so all the interactions were based purely on sound. A field study evaluation of this system showed that the high quality audio of the system worked well for supporting sociable exchanges. It did not provide a means to set up one-to-one private conversations; it only supported a multi-user

shared audio space. Additionally, users complained about having no way to tell who was currently logged into the system – Thunderwire did not have a way of representing presence, and users could not know who was available on the system without explicitly asking [10].

Cruiser is another system, implemented at Bellcore, which aimed to promote informal communication among members of distributed work groups. Unlike Thunderwire, Cruiser did not rely solely on audio, but added video as well, allowing users to initiate video conference calls with one another, using desktop workstations. Cruiser is an example of a media space, which was defined by Robert Stults of Xerox PARC as: "An electronic setting in which groups of people can work together, even when they are not resident in the same place or present at the same time. In a media space, people can create real-time visual and acoustic environments that span physically separate areas. They can also control the recording, accessing and replaying of images and sounds from those environments." [1, 17]

The Cruiser system attempted to mimic spontaneous face-to-face encounters between two individuals by randomly initiating video calls between selected users – these random system-initiated calls were called "autocruises." Though many users took advantage of the ability to initiate calls themselves, they very infrequently responded to autocruises, and in fact often cited them as being intrusive and one of their least favorite features of the system. Ultimately, Cruiser was used much like a telephone; it did not achieve its goal of supporting spur-of-the-moment informal communication akin to natural face-to-face encounters, as its designers were hoping [6].

Portholes, a project at XeroxPARC and Xerox EuroPARC, also used desktop workstations to support distributed group awareness, but instead of using video, Portholes displayed regularly updated images of public areas and offices in different workspaces. Employees at PARC in California and EuroPARC in Europe used the system to stay in touch and connected with each other. From a quick glance at the Portholes images, a user might get a sense of who else was around and/or available. Users needed only to glance at the image information to see what was going on elsewhere in the workspace; no further action was required. Though some users complained that the images were not updated frequently enough, and there was often not enough new information to ensure the

reliability of the system, Portholes was also applauded for saving users' time by allowing them to know whether certain individuals were in their offices, thus preventing unnecessary phone calls or office visits [4].

Hubbub, a project from AT&T Labs, is another system that aimed to promote opportunistic interactions and awareness among distributed groups. However, its installation as a mobile IM client did not limit its use to the workplace, making it similar to the SimPhony system, which is also mobile. One interesting aspect of the Hubbub system was its use of sound to identify the sender of each message. Each user was represented with a sound ID, or a short song clip, which was played preceding any message that was sent. Thus, without looking at the system, users could tell just by listening who had sent a message [11]. The Monkey Business system has a similar approach of using audio cues to identify which office a monkey is currently representing with its movements.

2.3 Physical Embodiment

One of the most obvious differences between the systems described above and Monkey Business is that in Monkey Business, we have physically embodied the information about other people and locations in the form of an animatronic agent. Physically embodied agents have several advantages compared to other types of awareness tools. First of all, physically embodied devices have the advantage of being able to display information in an ambient manner, blending into the physical environment and perceivable in the background of one's awareness [8]. An ambient display can portray nonessential information at the periphery of one's attention that will not compete with the more important information that one focuses on in the foreground [12]. Since one goal of this project is to display information in a non-intrusive manner, it makes sense to use a device that is less likely to disrupt one's primary area of focus. However, animatronic devices also possess the capability to make enough movement and sound to push their way into the foreground, if it becomes necessary to catch someone's attention. Thus, an animatronic device can be either ambient or attention-getting, as required by the situation.

Another advantage of physically embodied devices is that they are publicly viewable, and can convey information to several people at once [8]. Thus, for the purposes of this research, a physically embodied device may be preferable to a more private expression of information, such as a display on a computer screen or audio that is broadcast through headphones. Information presented on a computer screen is primarily intended to be viewed by the one person seated in front of the screen, and audio broadcast through headphones is only intended for the wearer of the headphones; in contrast, the information conveyed by an audio-equipped physically embodied agent is easily accessible to everyone in the room. Therefore, the agent enables greater ease of information sharing among co-located members of a group; for example, if several people are gathered in one office, the information that the agent communicates will be available to all of them at once.

Why did we choose a monkey in particular as the agent in this project? First, because the monkey has a face, arms and other human-like characteristics, it is able to interact with people in a somewhat human-like fashion. In her research on affordances of embodiment, Justine Cassell argues that using human conversational protocols in the design of an interface strengthens the function of the embodiment [2]. People also have a natural affinity toward cute, stuffed animals and are likely to anthropomorphize them.

Second, because we find the monkeys to be cute and lovable, we believe that they make appealing office adornments. Additionally, monkeys also have more intelligence than other animals. Thus, it seems logical that humans will trust monkey agents to convey intelligent information more than they might trust other animal embodiments. Monkeys also suggest an air of light-hearted playfulness. This light-heartedness complements the informal nature of communication that the system promotes. The choice of a monkey as an animatronic agent was largely personal, however, and if the agent proves to be a valuable tool in general, it is perfectly acceptable for it to assume different forms. Stefan Marti, in his research on physical embodiments, notes that diverse embodiments of agents are expected because users will exhibit individual preferences for different animatronic forms [18].

The TANGERINE system, developed at Lancaster University in the United Kingdom, is a bit like Monkey Business in that it also used a light-weight physical

embodiment as a means of notification; its goal was to enable distributed members of a workgroup to be aware of each other's presence in a light-hearted way, using a wooden, motor-driven parrot as a notification device. The researchers noted that many office workers already had toys and mascots on their desk; thus the office environment was not greatly impacted by the introduction of another toy-like device. The parrot was controlled by a webserver; remote users simply needed to press a button on a webpage, which caused the parrot to spin around. This enabled remote users to demonstrate their presence and informally express greetings to each other via the parrot. The researchers also added bird-tweeting sounds as an auditory alert of events, as well as introducing a text-to-speech engine with which to announce the name of the initiator of each event. Users appreciated the humor of using a parrot in conjunction with bird sounds as a means of notification, and this humor actually worked to diminish their annoyance at the potential disruptiveness of auditory alerts [16].

Other research has been done in the area of using different kinds of abstract displays to represent presence and activity. In the AROMA project, researchers at Roskilde University in Denmark experimented with representing the activities of remote others via four different types of ambient displays: an audio display playing the sounds of waves over speakers at different volumes, a handrest whose changes in temperature indicated changes in activity, a merry-go-round which rotated at different speeds, and a cloud animation in which the clouds drifted at different speeds. Overall, the researchers found that the users were able to use these displays successfully to get a sense of remote activity, though they had trouble deciphering the mapping of the changes in each representation to real-life events [20].

Chapter 3

System Details

3.1 Monkey Construction

Once we decided upon monkeys as the animatronic agent in this project, the first step was the construction of the animatronic monkeys themselves. Each monkey started out as a 10-inch hand puppet, as shown in Figure 3-1, with spaces available for a puppeteer's hand in the monkey's head and arms. In order to transform the puppet into an animatronic agent, we had to remove some of the puppet's internal stuffing and replace it with servo motors, stainless steel strips to hold the servos in place, and a microcontroller. The microcontroller was connected to a software program on a computer via a serial port, enabling the monkey to receive signals from the computer, which instruct each servo how to move. Figure 3-2 shows a view of the animatronic monkey with its body opened to reveal the servos, steel strips, and the microcontroller board inside.

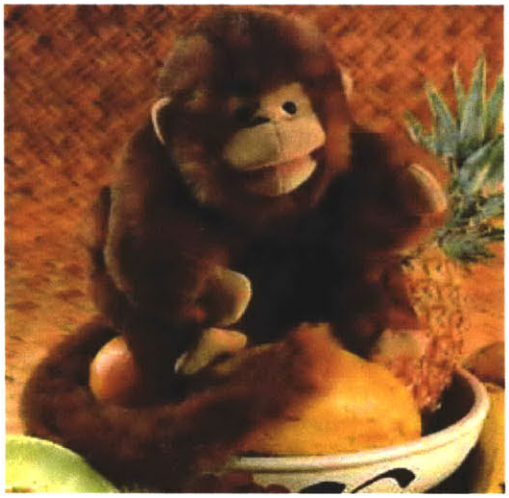


Figure 3-1. Original monkey hand puppet.

Figure 3-2. Monkey body open, exposing servos, rods and microcontroller board inside.

Each monkey has five servos, giving it a total of five degrees of freedom. The servos allow it to move its head up and down, to wave each of its arms back and forth, to move its body from left to right, and to swing back and forth from its tail. We decided to

construct the monkey so that it would hang upside-down from its tail, rather than sitting upright. In order to do this, we had to insert an aluminum rod into its tail, to make the tail stiff and strong enough to enable this hanging position. We also constructed a stand for each monkey to hang from. The stand is built from a chemistry lab support stand, a clamp, and a metal rod. The monkey hangs from the rod as shown in Figure 3-3.

Figure 3-3. Monkey hanging from monkey stand.

We used a Pontech SV203 servo motor controller board as the microcontroller for the monkeys, which is pictured in Figures 3-4 and 3-5. To this board, we attached not only the servos, but also the motion and proximity sensors. The board is powered by wall plugs, and has a serial port, enabling us to connect it to a computer via a long cable that extends from the monkey to the computer. This connection allows the monkey to communicate with any computer through its serial port. Although a wireless configuration would have been more ideal, this setup was chosen in part as a cost-saving measure, and in part to avoid having to continually recharge batteries. The monkeys, once connected to computers, are able to communicate with each other via their respective computers' network IP addresses.

Each monkey is also equipped with a microphone and a speaker. The speaker sits toward the back of the monkey's head and is connected via rainbow cable wire to a small one-watt audio amplifier kit, pictured in Figure 3-6. The amplifier receives its audio signal from the speaker (or line-out) port of the computer, enabling the monkey to use the computer's sound card as its source of sound.

The microphone, shown in Figure 3-7, sits inside the monkey's body; it originated as a small cellular phone headset, from which we stripped away all the external housing. We soldered the microphone to a rainbow cable wire, which we attached at the other end to an audio connector. The audio connector plugs into the microphone (or line-in) port of the computer.

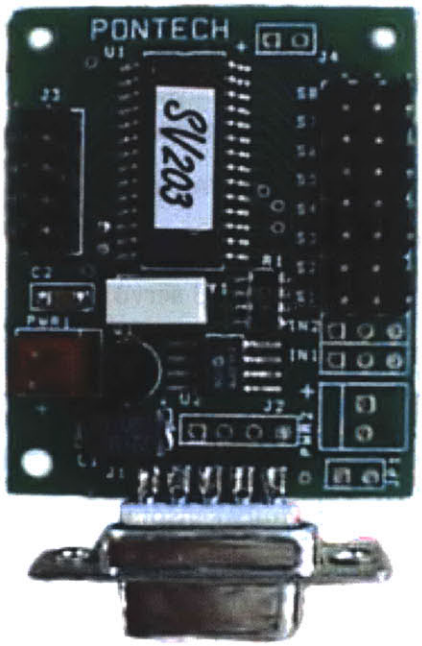


Figure 3-4. SV203 microcontroller board.

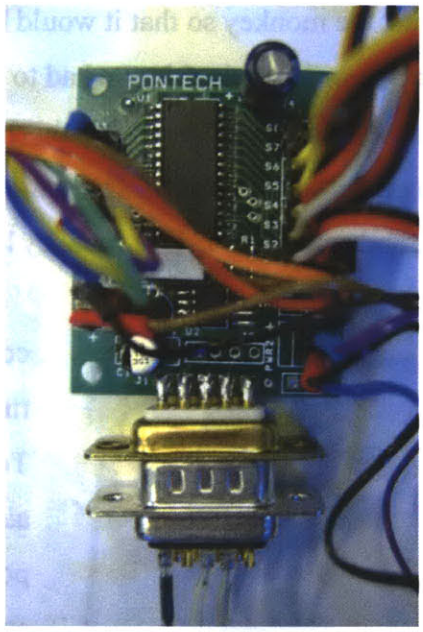


Figure 3-5. SV203 microcontroller board with servos, power, sensors, serial port plugged in.

Figure 3-6. Audio amplifier kit.

Figure 3-8. Sharp GP2Y0A02YK IR proximity sensor.

Figure 3-7. Microphone from cellular phone headset.

Figure 3-9. Model 442-3 IR-EYE integrated motion sensor.

Each monkey also has a Sharp GP2Y0A02YK IR proximity sensor and a Model 442-3 IR-EYE Integrated motion sensor, shown in Figures 3-8 and 3-9, respectively. These sensors are affixed to the monkey's stand, rather than being attached to a part of the monkey itself. The reason for this attachment to the stand is that the sensors are intended to sense the motion and proximity of people in the room that the monkey occupies. If the sensors were attached to the monkey itself, then whenever the monkey moved, the sensors would move as well and consequently would sense their own motion. Thus, in order to be effective, the sensors need to be fastened to something stationary.

Figure 3-10 shows the motion and proximity sensors affixed to the monkey's stand; the monkey's tail also hangs from the stand to the right of the sensors. Notice that there is a cone-shaped object around the motion sensor. This object is a small, plastic Fresnel lens; it reduces the noisiness of the data read in from the motion sensor, and thus makes the motion sensor data more reliable.

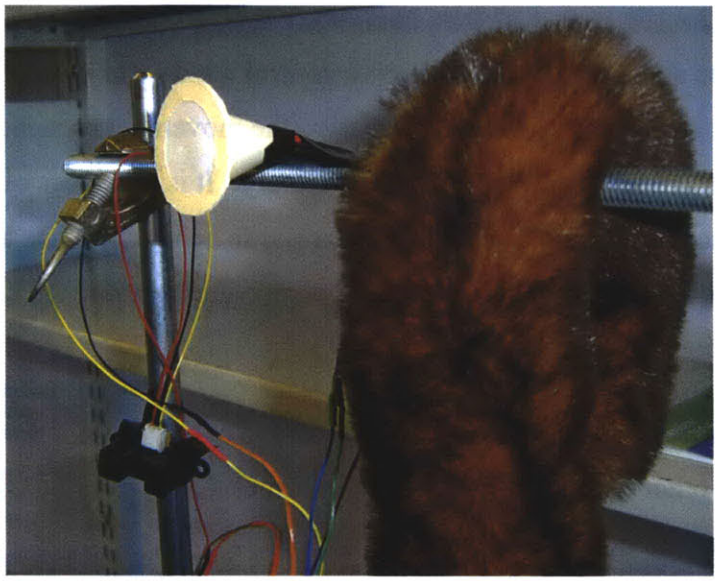


Figure 3-10. Motion and proximity sensors affixed to the monkey stand, from which the monkey is also hanging by its tail. The motion sensor has a cone-shaped Fresnel lens around it.

3.2 System Architecture

The monkeys are controlled primarily by an animatronics controller software application that was written in Visual Basic. This application communicates with the monkey through a computer's serial port. A screenshot of the application's interface can be seen in Figure 3-11. This interface allows the user to record different movement

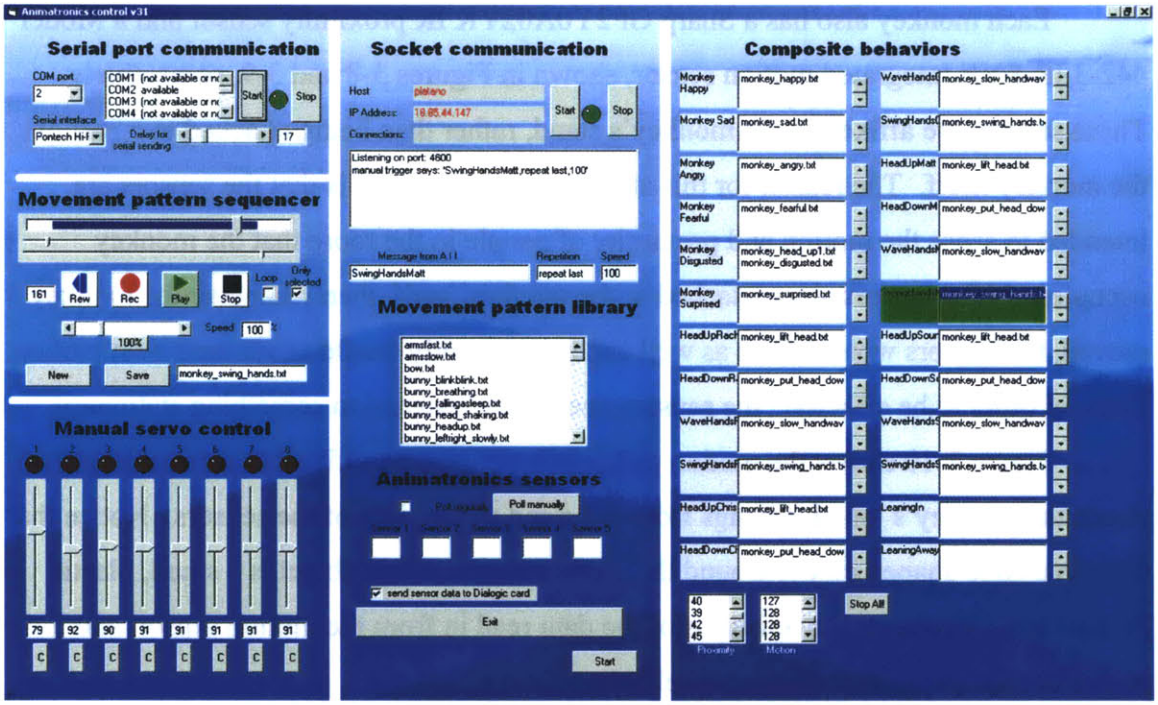


Figure 3-11. Animatronics control software screenshot.

patterns, using the sliders in the manual servo control window to control each one of the monkey's servos individually. The user can save the movement sequences he or she creates, and add them to the movement pattern library. It is then possible to combine several of these sequences and save them as composite behaviors. The application can play sounds associated with each pre-recorded composite behavior. A close-up of the composite behaviors is shown in Figure 3-12.

The animatronics controller application also performs several other functions. It listens over a socket port for any incoming messages that tell it which pre-recorded behaviors the monkey should execute. It also reads in raw motion and proximity sensor data from its monkey's sensors, processes the sensor data, and if it determines that a significant motion or proximity event has occurred, it sends this information to a server called the SuperMonkey. The SuperMonkey then transmits the event out to the network of other monkeys.

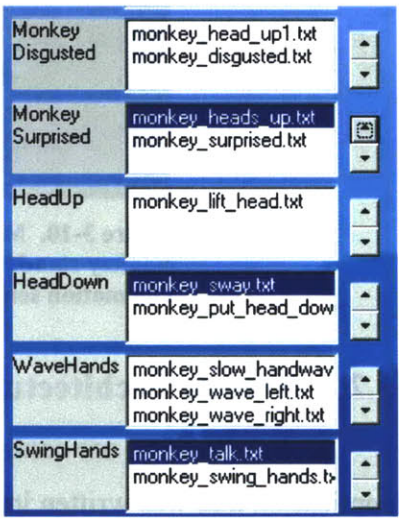


Figure 3-12. Close-up screenshot of composite behaviors.

Table 3-1 shows how the animatronics controller decides from the sensor data whether an event has occurred, as well as the corresponding behavior that a monkey exhibits upon receiving notification of an event. This behavior usually consists of a combination of movement and sound; the sound indicates where the event occurred, while the movement indicates which type of event occurred. The action that a monkey performs upon receiving a message from the SuperMonkey can actually be customized by the end user. In the current version of the system, however, all monkeys react identically upon receiving the notification of the same event.

Sensor Data	Event Sent To SuperMonkey	Corresponding Monkey Action
Proximity: Values range from 0 (furthest) – 135 (closest)		
Proximity sensor data value exceeds 80 and is at least 30 greater than the previous value	Approaching	Monkey lifts its head, and plays sound to indicate where the event occurred
Proximity sensor data value is less than 80 and is at least 40 less than the previous value	Retreating	Monkey puts its head down, and plays sound to indicate where the event occurred
Proximity sensor data value exceeds 95 for 3-5 consecutive seconds	Leaning in	Monkey plays a sound to indicate that the audio channel is opening, and starts streaming audio
Proximity sensor data value is less than 60 for 3-5 consecutive seconds	Leaning away	Monkey plays a sound to indicate that the audio channel is closing, and stops streaming audio
Motion: Values range from 0 (furthest left) - 256 (furthest right). Values usually hover around 128, which is neutral (no movement).		
Motion sensor data value is less than 112	Moving to the left	Monkey swings both arms in opposite directions, and plays sound to indicate where the event occurred
Motion sensor data value exceeds 144	Moving to the right	Monkey waves both arms in the same direction, and plays sound to indicate where the event occurred

Table 3-1. Sensor event processing and mapping to behavior.

The function of the SuperMonkey Server is very simple. Whenever a monkey's sensors detect an event, such as significant movement, or someone approaching or retreating from the monkey, the animatronics controller sends this event to the

SuperMonkey. The SuperMonkey then sends the event out to all the monkeys in the network, except to the monkey that detected the event in the first place. The Super-Monkey's place in the system architecture can be seen in Figure 3-13.

More detailed information about how events are processed can be found in Appendix A. This appendix includes flowcharts for each type of incoming message a monkey can receive (from the sensors, the microphone, or the SuperMonkey), and indicates how the monkey decides to react to each message.

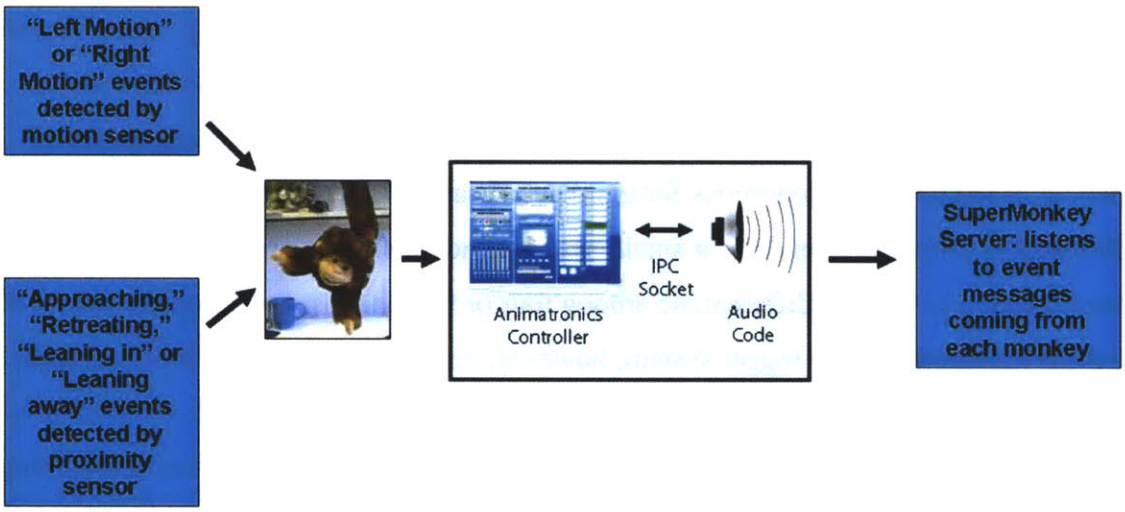
3.3 Audio

The final piece of the Monkey Business system is the audio component. There are two ways of playing audio in the system. One is through the animatronics controller, which can play short pre-recorded sound cues. The other is through a separate audio application written in C++. This application is installed on the same computer as the animatronics controller, and can receive commands from the animatronics controller through an IPC (inter-process communication) socket. It is primarily used to play "live" streaming audio from one monkey to another. The API for this code can be found in Appendix C.

3.3.1 Pre-recorded Audio Cues

There are two primary purposes for the pre-recorded sound cues. The first is to let the occupants of each office know where an event has occurred when their monkey starts executing a behavior. Each office has a sound cue associated with it. When a monkey receives notification of an event from the SuperMonkey, it also receives the location where the event took place. Thus, upon receiving an event message from the SuperMonkey, a monkey behaves in accordance with the event it received, and plays the sound mapped to the office where the event occurred. It is very easy to change which sound is mapped to which office; this information is specified in a configuration file that is read by the animatronics controller when it starts up.

In the current implementation, we are using edited snippets of recorded chimpanzee vocalizations as the sound cues associated with each office. Thus, it is necessary to learn which snippet is mapped to which office in order to know where each



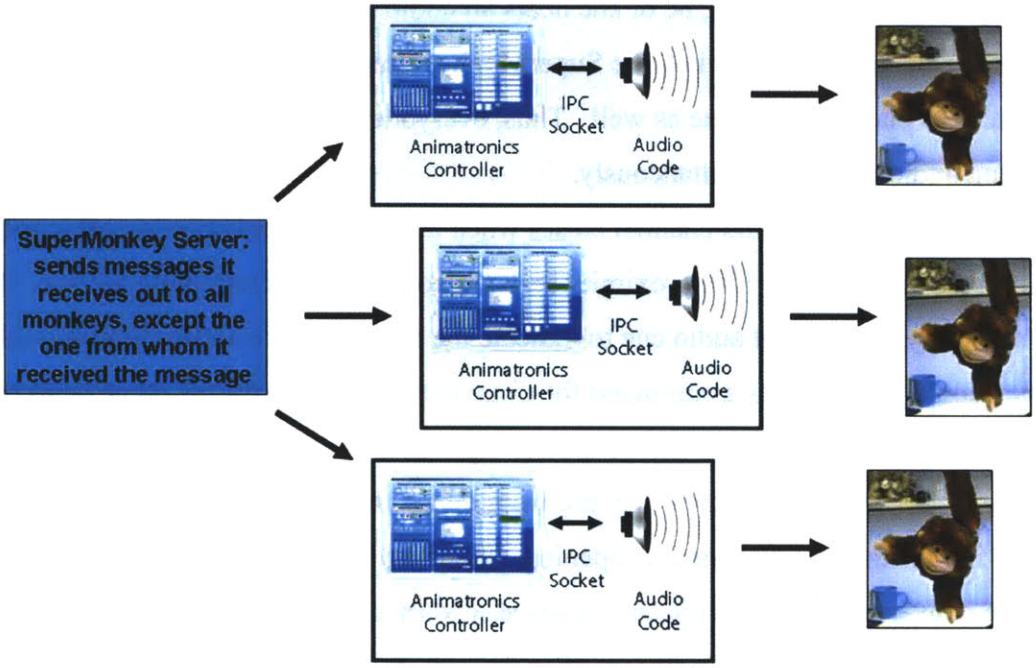


Figure 3-13. Flow of sensor event information from each monkey to the SuperMonkey Server, and then from the SuperMonkey Server back out to each monkey.

incoming event has occurred. It might be simpler for the system's users if we replaced the chimpanzee sounds with the sound of each office occupant saying his or her name aloud – this would make learning a sound-to-office mapping unnecessary. However, we believe it is more in character with the lightweight nature of the system to have the monkeys produce monkey-like noises as sound cues rather than articulating names of people; it seems a bit incongruous for monkeys to say people's names. Also, Monkey Business is primarily intended for small groups of no more than a few users, and it is not very difficult to learn to differentiate among four or five sounds, and associate each with a different location. In a bigger system, however, this approach might not be practical, as the learning curve of the sound-to-office mapping would become steeper.

The second use of pre-recorded audio is to notify system users when a streaming audio channel is about to open or close. One initiates the opening of a streaming audio channel by leaning in toward the monkey and remaining there for a few moments. This leaning in motion is detected by the monkey's proximity sensor. Once a user has been leaning in for a few seconds, he or she hears an audio cue indicating that the audio channel is opening. In addition, the SuperMonkey sends a command for all the other monkeys to play an audio cue as well. Thus, everyone in the system is made aware of the opening audio channel simultaneously.

Similarly, the audio channel closes when the user who initially leaned in moves away from the monkey. The proximity sensor also detects this retreating motion. The monkey plays a different audio cue to indicate the closing of the audio channel, and the SuperMonkey also issues a command for a cue to be played on all the other monkeys.

In this case, the audio notification is especially useful if system users are having a sensitive conversation that they may not want others to hear. When the sound plays indicating that an audio channel is opening, users may want to temporarily pause their private conversation. Thus, the cue serves as a warning that others can now hear the activities in each office in the system. For the person who initiates the audio chat, the sound cue serves to indicate the successful opening of the channel. The initiator can now start talking, and will be heard by other users in remote offices. Another cue plays to indicate the closing of the audio channel, which means that streaming audio is no longer being transmitted between offices.

We are currently using different cues to indicate whether the audio channel was opened or closed by a local or remote user. We use the AOL Instant Messenger audio cues for sending and receiving instant messages as the prompts for opening and closing an audio channel locally; these are the cues that the person who initiates and terminates the audio chat will hear. These sounds consist of three short tones played in ascending order for opening the audio channel, and descending order for closing the audio channel. Because users of AOL Instant Messenger are already familiar with these cues as signals for starting and stopping an IM chat, it made sense to borrow them here as signals for starting and stopping an audio chat.

When an audio chat is initiated, all of the remote users hear a chime sound as notification that someone else has opened or closed the audio channel. Much like the AOL IM cues, the chime's pitch ascends for the start of an audio chat, and descends when the chat has been terminated. We decided to use different indications for locally and remotely initiated chats so that users would know if they had started a chat accidentally, by inadvertently leaning close to their monkeys. This decision was made based on our initial tests of the system.

3.3.2 Live Audio Streaming

Once an audio chat has been initiated, we use UDP multicasting to stream live audio to all monkeys in the system. Through an IPC socket, the animatronics controller sends a command to the C++ audio application to start streaming live audio from the location where the audio chat was initiated to all other locations. Simultaneously, the SuperMonkey sends a "Leaning in" event message to the animatronics controller of every other monkey, which then instructs each monkey's audio application to start streaming live audio as well. Thus, now every location in the Monkey Business network can hear what is happening in every other location. Figure 3-14 illustrates how the UDP audio multicasting functions in a system architecture diagram.

After the audio chat has been started, if a second person leans in to his or her monkey in another location, the nature of the audio channel switches from a public chat, in which everyone in the system can hear everyone else, to a private chat, in which only the two people who are leaning in can hear each other. This is accomplished by

switching the audio chat to a different UDP address, to which only the two leaning-in parties are connected. If a third person then leans in, he or she joins the private chat between the first two people, making it a three-way chat; if the third person then leans away, the private chat is restored to the first two participants. The chat progresses in this manner, with people leaning in and away to join and leave the chat. When one of the last two remaining chat participants leans away from his or her monkey, and only one person remains, the chat ends and the audio channel closes for everyone. It does not re-enter the public state that it was in initially, in which everyone can hear everyone else.

We chose this design because it easily enables Monkey Business users to use the system for both public broadcasts to everyone, and for more private conversations that do not need to be broadcast to everyone. The system accomplishes this goal while still being intuitive to use; the system users only need to know that they must lean in toward the monkeys to be heard, and do not have to worry about any settings beyond that. Leaning toward a monkey also brings a user closer to the monkey's microphone, which improves the audio quality – this is a side benefit of this design.

It is important to realize that no conversation on the Monkey Business system is ever completely private, as anyone can join in at any time. However, because the goal of the Monkey Business system is to promote spontaneous group communication, we view this feature as a benefit, rather than a privacy invasion. Monkey Business users should be aware upfront that the system is designed with the intention of keeping them more connected, and that sometimes, it may do so at the expense of privacy. If a user is particularly worried about compromising his or her privacy in a sensitive situation, the best way to guard against this is to temporarily disconnect his or her monkey from the system. Monkey Business is aimed at users who want to have a relatively intimate connection with other members of their group, and who are thus not concerned about keeping things private from fellow group members. This notion of sacrificing privacy in exchange for better group communication is also important to keep in mind for the last audio feature I will discuss.

3.3.3 Listening In

As mentioned earlier in Chapter 2, Monkey Business borrows the notion of periodic eavesdropping from the ListenIN system, in which the system listens to bits and pieces of activity happening in remote locations. This is accomplished with further communication between the monkey's microphone, the animatronics controller, and the C++ audio application.

The monkey's microphone records samples of live audio from the office it occupies, each about five seconds in length. The animatronics controller application periodically calls upon a module, which analyzes these audio samples. If upon analysis, an audio sample is deemed to be significant, then using UDP multicasting, this audio sample is transmitted over the Monkey Business network to every monkey in the system, except the monkey that recorded it. The multicasting of the sound sample is slightly delayed from the time it actually occurred, because it needs to be recorded and analyzed before it can be sent out to the network.

Currently, we are using amplitude as the measure by which to determine audio significance – if an audio sample exceeds a certain amplitude threshold, it will be multicast to the network. In future versions of Monkey Business, we hope to implement a more sophisticated algorithm for determining whether audio is significant and should be broadcast. This may include listening for certain audio events, such as a telephone ringing, a door opening or closing, or voices having a conversation. But presently, our analysis methods are only advanced enough to use volume as a discriminating factor.

Thus, in addition to hearing live audio via the audio chats initiated by leaning in toward the monkey, Monkey Business users may also periodically hear short audio clips from other offices, when our analysis algorithm determines that an audio sample is noteworthy enough to be multicast. This additional function gives users an up-to-date awareness of what is happening in other offices, beyond the presence cues conveyed by the motion and proximity sensors. As mentioned in the previous section, privacy is a potential concern here, as users can never be sure when something they say might be broadcast to the whole network of users. But again, we assume that users of this system are willing to forego some privacy in order to reap the benefits of enhanced group

communication that the system provides. And as mentioned earlier, users always have the option to disconnect their monkeys.

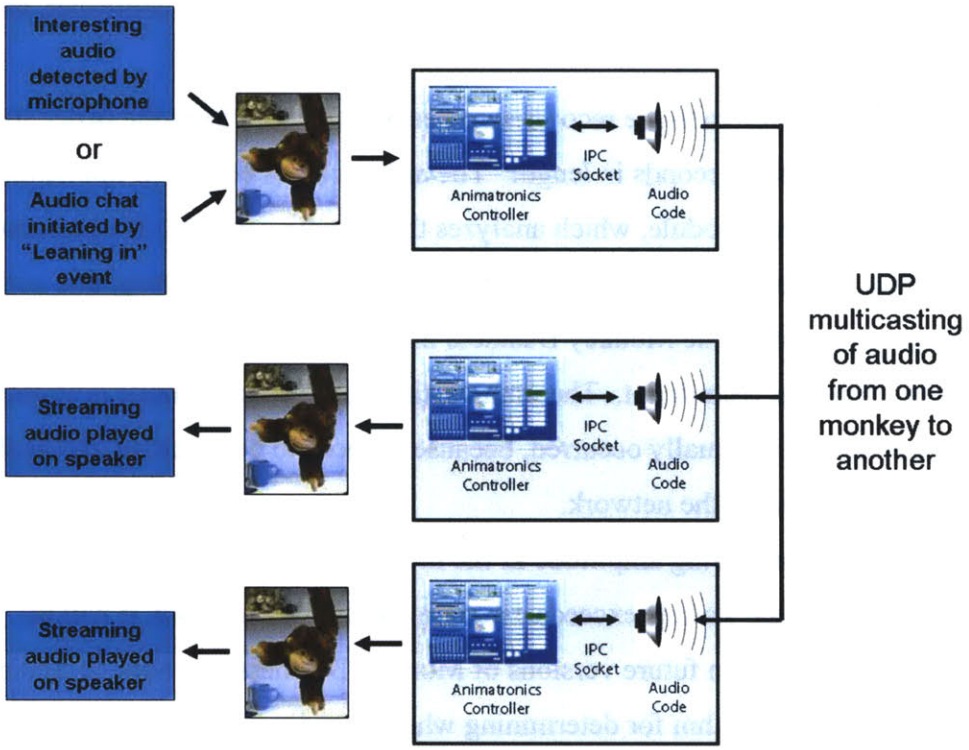


Figure 3-14. UDP multicasting of audio information from one monkey to the others. The SuperMonkey Server is not involved in the multicasting of audio streams.

Chapter 4

User Scenarios

The following section contains fictional scenarios to illustrate how the Monkey Business system might function in different settings. This section is intended to give the reader a more holistic picture of how the system could be integrated into everyday office or home life, and how the different features of the system might enhance group interactions in these settings. Some of the features described in these accounts have not yet been implemented, but this is how we envision the completed system to be functioning in the future.

4.1 Office

For the office scenario, I will re-visit the characters from the fictional example given in the introduction. In this example, Larry and Kate are colleagues who share an office, while their supervisor, Mark, occupies the office next door. All three are working together on a project along with a fourth team member, Susan, whose office is on the other side of the building. In order to enhance team collaboration and promote informal and spontaneous communication among all team members, the team has implemented a Monkey Business system, which is up and running. There is one monkey in Larry and Kate's office, one in Mark's office, and one in Susan's office.

One day, Susan and Mark are at work early, and neither Larry nor Kate has arrived yet. Mark leaves his office for a meeting. Susan's monkey plays the audio cue that signifies activity coming from Mark's office. Susan glances at her monkey out of the corner of her eye and sees it put its head down, hiding its face. She knows from the monkey's action that Mark has just left his office, and she continues working. A little while later, the monkey plays a different audio cue, signifying activity in Larry and Kate's office, and it lifts its head. Susan knows from this action that either Larry or Kate has arrived; she mentally notes someone's presence in that office, but the notification is not disruptive enough to interrupt her work.

A few minutes later, Susan encounters an issue in her work that requires input from Kate. Susan leans in toward her monkey. After a moment, having detected her leaning in motion with the proximity sensor, the monkey plays a brief sound, indicating that it has opened up a full-duplex audio connection with the other monkeys. All of the other monkeys also play the same sound, to let their office occupants know that an audio connection has been established, and that anything they say now will be broadcast into the other offices through microphones in the other monkeys. Susan says: "Kate? I need your input on the McGraw file."

Larry hears Susan through the monkey in his office and replies to her: "Hi Susan, Kate is not in yet. But when she arrives, I'll let her know you're looking for her." They both move away from their monkeys; the monkeys again use the proximity sensor to perceive this retreating motion. The monkeys in all offices play another sound to indicate that the audio connection is now closing.

Mark's meeting ends, and he returns to his office. Susan and Larry are made aware that Mark is back in his office when their monkeys both play Mark's cue sound and raise their heads. Susan and Larry's monkeys also wave their arms around, having detected additional motion in Mark's office. This might indicate that Mark is moving around a lot, or perhaps someone has come back to his office with him, causing extra activity in his office.

Susan's monkey then plays the cue sound from Larry and Kate's office and raises its head again. Susan realizes that this might mean that Kate has arrived, but she has another task to finish before she returns to the task that requires Kate's input. A few minutes later, however, Susan hears a snippet of conversation from Larry and Kate's office through her monkey. Larry and Kate's monkey, which has been periodically recording and analyzing the audio in their office, has deemed this particular excerpt as significant in its analysis. Thus, it has sent the audio clip out to the monkey network, to be played through all the other monkeys. There is a slight delay in between the live occurrence of the audio, and its replaying on all the other monkeys, but the delay is not significant enough to render the sound snippet irrelevant by the time it is played in the other offices.

In the sound snippet, Larry tells Kate that Susan wants her input on the McGraw file; Kate responds that she is not yet done with her analysis, but plans to get it to Susan later that afternoon. Susan, upon hearing this, leans in toward her monkey to initiate the full-duplex audio connection. Everyone is made aware of the audio connection establishment through their monkeys' audio cues. "Hi Kate and Larry, I overheard you guys talking," Susan says. "Kate, it's fine if you finish with the McGraw file later this afternoon, just let me know as soon as it's ready for me."

Mark, who has also heard the conversation through his monkey, decides to chime in. "Hi team. I actually just had a meeting about the status of the McGraw case. Everything has been delayed, so you all have a couple more days to work on the project."

"Oh that's great news, now I can catch up on some other work," Susan says.

"Susan?" Kate says, leaning into her monkey. "I actually have a question for you about the analysis." As Kate and Susan continue talking through their monkeys, Mark's audio connection to the conversation closes, since his monkey senses that he is no longer participating. Now there is a private audio channel between Susan's office and Kate and Larry's office. When Susan and Kate finish their conversation, they both move away from their monkeys and the private audio channel closes.

4.2 Home

Monkey Business was initially designed and tested with an office or work environment in mind, but as a result of building and using the system, we thought it might also work well in a home setting, to be used among family members as opposed to colleagues. Several comments from our user studies also point to the potential success and relevance of Monkey Business in the home. Thus, in this fictional scenario, I describe how Monkey Business might enhance home and family life.

Karen and Mike are married, and have three children, Sam, Eliza, and Jennifer. Karen often travels as part of her job, but likes to keep in close touch with her family while she is on the road. She has a portable monkey that she can take with her when she travels; her family also has the Monkey Business system set up and running in their home. There is a monkey in each of the three children's bedrooms, one in Karen and Mike's bedroom, one in the kitchen, and one in the family room.

One evening, Karen returns to her hotel room after a long day of meetings with clients. She has set up her monkey in her hotel room, and as she enters the room, she notices that her monkey is very active, frequently raising its head and waving its arms. It is playing audio cues that indicate that it is currently representing activity occurring in the kitchen of Karen's home. Karen realizes that it must be dinner time back at home.

Karen's monkey plays the audio cue indicating that it is about to open a full-duplex audio connection. A few seconds later, she hears Eliza saying, "Hi Mommy, we just had dinner. Daddy made us spaghetti."

"Hi everyone," Karen says through her monkey. She catches up with her family as they clear the table and wash the dishes. Eventually, they are finished cleaning, and everyone starts leaving the kitchen. Karen's monkey plays an audio cue that it is closing the audio connection.

After dinner, Karen receives notification through her monkey that someone has entered Jennifer's bedroom. Her monkey also indicates that there is activity going on in the family room. She suspects that her husband, Mike, is probably watching the news on TV in there. However, a few minutes later, she hears the theme song from Sam and Eliza's favorite television show playing through her monkey. The family room monkey's periodic audio analysis of sounds in the room has determined that this sound snippet is significant enough to be multicast over the network. Karen leans in toward her monkey to initiate an audio connection.

"Sam, don't you have a spelling test tomorrow? I think you should be studying for it, not watching television," Karen says. "And I believe that Eliza has a book report she needs to work on."

"Sam, is your test tomorrow?" Karen hears Mike asking in the background. "I thought it was on Friday." Mike then leans in toward the monkey to talk to Karen. "Hi honey, I'll make sure that Sam prepares for his test."

"But Daddy…" Karen hears Sam complaining. A few minutes later, Karen's monkey indicates that people have left the family room, and now instead, there is activity in both Sam's and Eliza's bedrooms. Later on in the evening, Karen's monkey is less active and gives no more indications of activity in any of her children's bedrooms. She realizes that her children must have gone to sleep. Karen feels comforted that she can

still maintain a peripheral awareness of what is going on with her family at home, even though she can't physically be there with them.

Chapter 5

Evaluation

We evaluated the Monkey Business system using two different methods. First, we conducted a pre-planned formal user study that we ran over a period of four weeks; the participants in this study were not otherwise involved with the Monkey Business project. We ran this study before completing implementation of the system, in order to gauge users' reactions to having animatronic monkeys in their offices over an extended period of time. We hoped to use the feedback from this study to inform our design of the system.

Our second method of evaluation was more informal – we used the Monkey Business system ourselves, in the offices of members of our own research group, during different phases of implementation. This helped us not only with debugging the system, but also in deciding which features we liked and found useful, and which others were merely annoying rather than informative. It also enabled us to envision other features that we might like to add in the future.

5.1 User Study Design

We ran our more formal user study over a period of four weeks. We had six subjects in total, including two administrators and four students. Our first two subjects served as our pilot subjects; we slightly modified the study based on feedback from these two subjects. Each subject had an animatronic monkey placed in his or her office for a period of three to five days, which was long enough for the initial novelty of the monkey to wear off. The monkeys were not yet equipped with sensors or microphones at this point, so nothing that the users did or said was recorded in any way.

The primary experimenter set up a small Java application that allowed her to control the monkeys remotely by sending behaviors from her office to the monkeys in the subjects' offices. She sent behaviors to the monkeys throughout the day while the subjects were in their offices, at a rate of approximately one behavior every ten to fifteen

minutes. The behaviors consisted of a mixture of movement and sound; in between each behavior, the monkeys remained static and silent.

Brief online surveys were also sent to the subjects directly following the sending of a monkey behavior, although surveys were not sent following every behavior. Rather, subjects received about one survey an hour. So during a typical hour of the study, a subject could expect to see the monkey move about four to five times, and receive one survey following one of these monkey behaviors. The surveys were meant to gauge each subject's reaction to the preceding behavior. They collected information such as what the subject thought the monkey was trying to express through movement and through sound, and whether the monkey was too disruptive.

We designed the surveys using the Experience Sampling Method (ESM) [15]. The goal of the ESM approach is to record what the subject is doing and feeling at intermittent times throughout the course of the study. With enough subjects and samples, the data collected using this method can be used to build a statistical model of activities. The ESM results are also less subject to errors in recollection, as opposed to data collected after the study is over. The questions for these ESM surveys can be found in Appendix B.

There were two variables, each with two conditions, in the behaviors that were sent. The first variable was what the monkey's behavior represented. In the first condition, the monkey's actions represented the level of overall activity that was taking place elsewhere. For example, the monkey moved only slightly to indicate a low level of activity, but moved with greater force and gusto to indicate a greater amount of activity. Additionally, the volume of the monkey's sound was adjusted to be either lower or higher to reflect the current level of activity, i.e. greater volume corresponded to a greater amount of activity. For this condition, we used only one sound, which was a generic background noise of people talking and moving around; however, no specific dialogue could be distinguished.

In the second condition of this variable, the monkey's actions represented a specific activity. The specific activities we chose to represent included entering the room, leaving the room, talking on the phone, having a conversation with another person in the room, and typing at the computer. We programmed different monkey movements

to represent each of these activities, and also recorded corresponding iconic sounds. For example, for the activity of typing, the monkey would look up and move its arms rapidly, while playing the sound of keys tapping on a keyboard. For the activity of leaving a room, the monkey would lower its head to hide its face, while playing the sound of a door shutting. The full spectrum of the movements and sounds used in the user study can be found below in Table 5-1.

Activity	Description	Sound (if included)
Level Of Activity		
Level 1 (least amount of activity)	Monkey lifts its head up and shows its face. After a few seconds, it lowers its head.	Background noise at low volume
Level 2	Monkey looks up and moves both arms slowly. After a few seconds, it stops moving its arms and lowers its head.	Background noise at low to medium volume
Level 3	Monkey looks up and moves both arms rapidly. After a few seconds, it stops moving its arms and lowers its head.	Background noise at medium to high volume
Level 4 (greatest amount of activity)	Monkey looks up, moves both arms, and swings back and forth by its tail. After a few seconds, it stops moving and lowers its head.	Background noise at high volume
Specific Activity		
On the phone	Monkey nods its head and waves its arms for a few seconds, but keeps its head down. The lowered head represents unavailability (as a phone call is a private conversation).	A phone ringing three times
Having a conversation with someone else in the office	Monkey nods its head and waves its arms for a few seconds, and also looks up. The raised head represents availability (as an in-person conversation is usually public – others can join in).	Background noise (indistinguishable dialogue) at medium to high volume
Entering the office	Monkey lifts its head and shows its face to indicate availability.	A door opening, repeated twice
Leaving the office	Monkey lowers its head to hide its face to represent unavailability.	A door slamming, repeated twice
Typing	The monkey looks up and moves its arms rapidly, as though it is "typing."	Intense keyboard typing sounds for several seconds

Table 5-1. User study monkey behaviors and sounds.

The second variable was simply whether or not sound was included in the behavior. In one condition, sound was a part of the behavior, and in the other condition, there was no accompanying sound to each behavior. The two variables with two conditions each resulted in a total of four possible combinations of monkey behavior:

- Level of activity without sound
- Level of activity with sound
- Specific activity without sound
- Specific activity with sound

Each subject experienced at least three of the four combinations. After conducting the first two pilot studies, we eliminated the "specific activity without sound" combination, as subjects were having trouble understanding it and differentiating it from the "level of activity without sound" combination.

As much as possible, we tried to ensure that all the behaviors sent on a single day of the study were made up of the same variable combination. When this was not possible, we grouped the combinations into periods of time, so that all the behaviors sent within one time period had the same combination. We wanted to see if the subjects noticed the differences between the combinations, and which they thought were the most and least intuitive and disruptive.

After each study was run, we also conducted an informal, 30-minute interview with each subject. This post-study interview augmented the data that had already been collected using the ESM technique. A list of the questions that we asked each subject during these interviews can be found in Appendix B.

5.2 User Study Results

Although we had a fairly high response rate of about 80% to the ESM surveys that we sent out during the study, we realize that we cannot attribute a great deal of significance to these results for several reasons. First, in order for the quantitative data from these surveys to have more significance, we would need to have a larger sample size. Given that the first two subjects served as pilot subjects and that the user study was modified for the remaining subjects based upon the pilots' experiences, we really only had four subjects from whom we collected valid data. Second, due to complex

scheduling issues, each of these four subjects experienced having the monkey in their offices for varying amounts of time, which may have led to inconsistent results. Lack of time and resources prevented us from running a larger and more organized user study. Nevertheless, the most interesting results from the surveys are presented below.

All of the subjects (both pilot and regular) were able to provide us with insightful commentary on their experiences with the animatronic monkey in their post-study interviews. These post-study interviews proved to be a valuable source of rich qualitative information, and are also summarized below.

5.2.1 Quantitative Results

The quantitative data collected from the surveys yielded a few interesting results. It showed us that overall, of the specific activities, the subjects found that the monkey's representation of someone having a conversation with someone else in the room was the most intrusive activity, when sound was also played. The least intrusive activity was the monkey's representation of someone leaving the room. For the level of activity condition, surprisingly Level 3 was considered a bit more intrusive than Level 4, but predictably, Level 2 was found to be less intrusive than Level 3, with Level 1 being the least intrusive of all. When subjects only rated intrusiveness in terms of sound and not movement, Level 4 had the most intrusive sound, and Level 1 had the least intrusive, as expected. For the specific activities, the conversation sound was the most intrusive, while the leaving sound was least intrusive – this result also corresponds overall to the specific activity the subjects found most and least intrusive.

Entering and leaving were considered the most intuitive of the monkey's expressions of specific activities, when sound was included. Having a conversation was rated as least intuitive. The representations of level of activity were found to be less intuitive in general than those of specific activities. The subjects had a much harder time understanding the meaning of the volume changes in the level of activity sounds than they did in interpreting the iconic sound cues in the specific activity sounds.

5.2.2 Qualitative Results

I have grouped the qualitative results into four categories, into which most of the subjects' comments fell. These categories include novelty factor, expressiveness of movement and sound, affinity, and suggestions.

Novelty Factor

Most of the subjects reported needing only a brief period of time to get used to having the monkey in their offices; many said that they felt they were used to having it there by the second day of the study, whereas one subject said it only took a couple of hours to adjust to the novelty of the monkey. Another subject said: "The more time passed, the more comfortable I was with the monkey in my office." This sentiment was echoed by some of the other subjects as well, who reported that their comfort level with the monkey gradually built up over time.

Only one subject found that the novelty did not wear off during the course of the study, but felt that she would get accustomed to it, given more time with it: "I still look at the monkey every time it moves. Three to four days is not enough for the novelty to wear off. But I would probably get used to it eventually."

The presence of visitors to the office re-introduced the novelty of the monkey to a certain extent: "Whenever there was someone else in the room and the monkey moved, the conversation turned toward the monkey. He was much harder to ignore with other people in the room."

However, this effect also wore off a bit as time passed: "When other people were there, they would talk about the monkey at first, but after awhile, other people ignored it."

"[The presence of other people] didn't have much of an effect. At the beginning, the monkey was a topic of conversation between my officemate and me."

Expressiveness of Movements and Sounds

The subjects had several different reactions to the movements and the sounds that the monkey made. In general, the subjects found that the movements alone were hard to interpret, but that the addition of sounds often clarified the monkey's expressions. One

subject commented that on the second day of the study, the monkey made the least amount of sense because it had no sound. However, on the third day of the study, when the monkey made iconic sounds to represent a specific activity, this subject commented that the monkey made the most sense. He said, "The sounds were definitely helpful. The task noise was really clear."

Other subjects also appreciated the clarity that the sounds lent to the monkey's expressions. Some of their comments were: "The sound represented what was happening, a door opening or closing, or people speaking."

"The sounds helped clarify the expressions, because the movements were a bit vague. You could learn the types of movements the monkey was making, and maybe attribute those to certain messages, but the sounds helped with that."

Subjects had varying responses to the movements. Some subjects noticed that the monkey's movements differed from each other, and had different purposes: "The different times that [the monkey] moved were different behaviors. Some movements were just a reminder that he was there… [they] were more ambient. Other movements were an urgent call to attention. The arm movements were the most attention-getting, with his head up. The length of time of the movement was less relevant than the movement itself."

One subject was able to differentiate between the two conditions of the behavior variable: "There were two models: one was indicating the scale of something, importance or salience of information. The other was explicit behaviors for different categories of things, such as typing versus talking."

Other subjects did not notice as much of a change in movements and had a harder time attributing meaning to them: "Movements seemed pretty consistent. It varied depending on what type of message [the monkey] was sending… but movements stayed the same."

"I didn't find the movements to be meaningful. I expected the movements to tell me what the person was doing. Either [the monkey] moved or it didn't move. How long it moved was the only variable. The position itself didn't mean anything to me."

One subject usually listened to music on headphones when he was working in his office, so his experience with the sound and the movement was a bit different from the

rest of the subjects: "At first, when I saw the monkey moving, I would take off my headphones to hear the sound. But when I learned to recognize the movement, I no longer needed to remove the headphones, since the sound went with the movement. Then it didn't matter that I was wearing headphones."

Two of the subjects mentioned that the squeaking sounds made by the servo motors detracted from the overall effectiveness of the sounds, and that it would be better if these servo noises could be reduced: "The motors were so noisy that it made it harder to ignore his subtle movements... I heard the motors every time he moved.... The motor sounds mask the significance of the monkey's [own] sounds."

For the most part, subjects did not understand the significance of the background noise being played at different volumes to represent different levels of activity. In fact, many subjects found it frustrating that they could not discern what was being said: "For the cocktail party noise, I couldn't understand what was being said, it was almost ghostly, I was trying to understand but couldn't."

"I couldn't differentiate between the different 'people talking' sounds."

"I thought it was live sound samples from someone else's room... It was too noisy to work out what it really was. It might have been office sounds, but it was hard to say, kind of a background noise. Once I decided it was people speaking, it made a lot of sense, but that was more of a logical deduction than really knowing what it was."

Affinity

Regardless of whether the subjects understood or found meaning in the monkey's expressions, they consistently reported that they liked the monkey and enjoyed having it in their offices. The monkey's inherent cuteness and likability seemed to lessen the annoyance of any distraction it may have caused: "He was a cute little monkey, therefore his movements were endearing. I was never annoyed at the monkey because he was cute."

"I loved having him in the office. He lightened things up. When we had a meeting, seeing the monkey move made everything much nicer. It was a happy time."

"I enjoyed having it as something else in my office to interact with, to have my attention. It was a nice distraction from work. I like the idea of a monkey; it's fun. I'd say overall, I enjoyed the experience."

"If I had the monkey longer, I think I would get emotionally attached to [it]... because it's more human-like, it's an animal. It's part of the environment… it's like a pet."

Suggestions

Many of the subjects gave us suggestions for how they thought we could improve their interactions with the monkey and make it more useful. One common theme that emerged was that perhaps the monkey was better suited to an environment other than an office setting: "An obvious use of the monkey is to have a sense of presence with [a] significant other, parent, or someone you're close to. [The] monkey might not be the best tool for monitoring other people, like a manager keeping tabs on employees. [It might be] better for parent-child, boyfriend-girlfriend, husband-wife."

"If I were to use this, I would use it more for knowing what's happening in my house in Mexico. If my sister enters the house, then I would like [the monkey] to generate the sound of her voice."

"It was an interesting context – right now, I think everybody's pretty isolated in this office building, it's nice to have another way to communicate. I don't know if there's a better setting. It would be good in places where there's low social interaction. Maybe [I'd want to use the monkey] with friends, or a long-distance relationship…"

Another suggestion was to make the monkey customizable to an individual's preferences for sound and movement: "I would like to have pre-defined positions of the monkey. Assign those positions to different events (even the weather). Move the monkey into a position, and then say, I want to map that position to hot weather, for example… it would be cool to allow people to map the sound they want. Customization is very important."

One subject suggested that he should be able to pick who the monkey represents, depending on who he is looking for, or waiting for, at any given moment in time: "A lot of times, I'm waiting for [my advisor]… if I could select a small group of people [for the

monkey to represent] it would be useful, otherwise I forget who I'm looking for. I'd have to pick where the target room is."

Finally, two of the subjects proposed that the monkey's movements should be more ongoing, rather than stopping and starting: "It would be useful if it could do some subtle, micro stuff at all times, so it's constantly moving at least a little. This gives more of a continuous window into a person's life. The on-off quality of it was one of the weirdest things, since it's not like the other person [that the monkey represents] turns off."

"It would be cool if it's in a continuous level of moving and showing information. Sometimes the monkey seemed to be dead. It would be cool if he was always alive."

5.3 Informal Evaluation

The purpose of our informal evaluation of Monkey Business was twofold: first, to test how well the system worked from a technical standpoint, and second, to assess how useful the system was to our group, and what features we liked and disliked from a usability standpoint. Because we implemented the detection of events using sensors before we set up the streaming audio components, we evaluated the sensors and corresponding movements first, before turning our attention to the audio function.

5.3.1 Sensors and Movement

After we had constructed all the monkeys, and written the code for the SuperMonkey Server, the first part of the system that we successfully implemented was event detection by the motion and proximity sensors. The proximity sensor could detect two types of events: approaching and retreating. Likewise, the motion sensor could also detect two types of events: left-to-right motion, and right-to-left motion.

We had to assign different monkey movements to each of these events. The default state of the monkey was to have its head down, facing away from the user, with its arms at its sides. For the approaching and retreating events detected by the proximity sensor, we decided to have the monkey move in the same way that it did to represent "entering the office" and "leaving the office" in the user study, since these activities had been found to be the most intuitive, and are similar in concept to approaching and

retreating. Therefore, to represent the "approaching" event, a monkey lifts its head up to face the user, and to represent the "retreating" event, a monkey puts its head back down, hiding its face from view.

Experimentation with the motion sensor data revealed that the difference between right-to-left motion and left-to-right motion was very subtle. The difference was also not very meaningful in a small office space, especially if someone was seated right in front of the monkey. In this case, any normal subtle movements would often trigger a motion event, but we found that which event was triggered was inconsequential. The fact that motion had been detected at all was meaningful, as it indicated presence, but whether someone had shifted slightly to the left or to the right did not really tell us anything important. Thus, we decided to have only one behavior to represent either type of motion, which was simply that the monkey would pick its head up and wave its arms.

After running the system in this manner for a while, one of the members of the group commented that he would like his monkey to have more variety in its movements. Therefore, we decided to modify the motion behaviors slightly to achieve this. We changed the behaviors so that for the left-to-right motion event, a monkey waves both of its arms in the same direction (which was the original behavior for both motion events), and for the right-to-left motion event, a monkey waves each arm in opposite directions. The difference is subtle enough that there is little likelihood of anyone ascribing different meaning to the two behaviors, but it accomplishes the goal of having more variety.

There was still another problem with the monkeys' behaviors for representing motion events. In both cases, a monkey would pick its head up and wave its arms in either the same or the opposite direction. However, when the monkey was finished with this action, it never put its head back down. This seemed to indicate that it had detected an approaching event instead of a motion event. We realized that the monkeys had to return to the default position of having their heads down when they were finished with any action other than approaching. Thus, we modified both of the motion behaviors to finish with the monkeys lowering their heads.

We noticed after running the system for a few hours that the monkeys were often detecting ghosts; in other words, they were indicating that motion or proximity events had occurred in offices that were currently empty. After looking at the incoming sensor

data whenever a "ghost" was detected, we realized that the sensors were extremely sensitive, and would often react to wind or changes in lighting, in addition to people. Thus, we had to modify each event to be triggered only in the case of more extreme movement, in order to reduce the number of false alarms that the monkeys were sending. We were not able to eliminate the false alarms completely, but we did reduce them significantly. Eliminating them completely would probably require the purchase of much more expensive, finely tuned sensors.

5.3.2 Audio

We had several technical difficulties in getting the audio components to work with acceptable sound quality. We experienced a range of problems, from the UDP protocol dropping audio packets, to starvation of the audio buffer, which produced a scratchy staticky sound, to high latency during transmission, which caused too much of a delay in audio transmission to be able to have conversations. Additionally, the simultaneous buffering of audio for analysis and transmission of streaming audio via a lossy protocol produced quite a few problems in audio quality. We tried different techniques to overcome these difficulties, including experimenting with different packet sizes, writing code to check for dropped packets, and replaying the previous packet whenever we encountered a packet that was filled with corrupt data.

We also encountered a feedback issue with the streaming audio. Because the speaker and the microphone are placed close to each other inside the body of the monkey, when I spoke to another person through my monkey, I was able to hear my own voice playing through the speaker in the other person's monkey. Hearing this feedback distracted me as I was trying to talk. This is the classic speakerphone problem. One possible solution to this problem would be to use half-duplex mode for audio chats, instead of full-duplex mode, such that when one person starts speaking, the microphone in the monkey on the other end stops transmitting. However, this prevents users from being able to interrupt each other naturally. Another possible solution would be to use a separate desktop microphone for speaking that would be placed a certain distance away from the monkey, instead of using the microphone inside the monkey. Unfortunately, this solution is also not ideal, because a person cannot simultaneously lean in toward his

or her monkey, while speaking into a microphone located elsewhere. We will need to further experiment and refine our system design to come up with a better solution for this problem.

Another issue we encountered with our audio system was an unexpected interaction between the audio analysis code and the pre-recorded audio cues played on each monkey's speaker. Because of the proximity of the monkey's speaker and microphone, any audio played on the speaker would exceed the amplitude threshold set by the audio analysis module. Thus the audio analysis code would always determine that any five-second audio sample containing a pre-recorded audio cue was significant. Consequently, we had to modify the code to only analyze audio when an audio cue had not played over the monkey's speaker within the last five seconds. We also had to ensure that audio analysis did not take place when a streaming audio chat was in progress.

Upon the initial testing of transitioning from a public audio chat including all Monkey Business users, to a private chat including only two users, we realized that the monkeys that were not included in the private chat were still streaming audio to the public channel. To solve this problem, we simply sent out a message from the Super Monkey to all monkeys to cease streaming audio to the public channel, before the private chat between two users commenced.

One user in our research group complained that the monkey was much too noisy in general, and that its noises were more disruptive than its animatronic motion. This observation became particularly apparent when he was interacting with people in his office, or talking on the phone. He requested that the monkey not make sounds every single time it received an event message, especially if the monkey was receiving multiple event messages from the same location. We implemented an enhancement so that each monkey would only make a sound to indicate the location of an event if it had not made that particular sound within the last twelve seconds. We chose the value of twelve seconds arbitrarily, and currently this value is hard-coded into the animatronics controller code. However, the code can easily be changed so that this value is configurable by the end-user. This would enable each user to specify how frequently he or she wants to hear monkey sounds to indicate event locations; this setting would likely vary from user to user.

We discovered one major issue with using UDP multicasting, namely that multicast must be supported on every segment of a network in order to work. In our case, the Media Lab network is set up to reject packets sent to a multicast address. Thus, whenever our audio data has to make hops over multiple routers, it is blocked and does not reach its destination. This problem affects both the streaming audio chat as well as the audio analysis code, which enables users to "listen in" to significant audio in other locations.

To work around this multicast problem, we had to turn to alternate strategies to transmit audio between locations that were more than one network hop apart. We solved the problem for the audio analysis by saving .wav files that were deemed to be significant to a shared network drive, which was accessible to all computers in the system. Every monkey in the system could access and play these .wav files when instructed to do so. We have yet to implement an alternative approach to multicasting streaming audio; this problem is left for future work.

Chapter 6

Conclusions

6.1 Discussion

This work raises many issues for discussion, especially because of the highly controversial and unconventional idea of using animatronic agents as a means of increasing group awareness among a group's distributed members. One issue that we realized might be problematic early on was the challenge of attempting to create different monkey behaviors that would map to different events in an intuitive fashion. Though the monkey has five degrees of freedom, and therefore a large number of possible movement combinations, it was very difficult to ascribe an obvious meaning to each different movement. The user study confirmed that the subjects had difficulty in interpreting the meanings of the movements, unless they were accompanied by evocative sounds. The lack of expression in the monkey's face further exacerbated this problem.

This leads us to question whether an animatronic agent is the best tool to use for intuitive event mapping. It seems better suited simply to represent presence, and this is how we ultimately ended up using it ourselves, in addition to using the monkey's audio capabilities. This problem also leads us to believe that enabling greater end-user customization – allowing users to choose their own mappings of events to behaviors – is the most effective use of the agents. Even as system designers, we feel that we cannot design an event-to-behavior mapping that will appeal and make sense to all users, and thus it is best to let users do this themselves. This approach might also be more fun for the end users.

Another problem we encountered frequently is that in multi-person offices (or in common spaces), one monkey simultaneously represents the presence of multiple people. Thus, when one person enters a two-person office, this might let other members of the group know that someone is now in that office, but it doesn't inform them specifically of who is in that office. Perhaps a one-monkey-per-person model would be more effective than our current model of one-monkey-per-office. The advantage of our current model is that it preserves some privacy; for example, if a user shares an office, he can take comfort

in knowing that other users of the system will not necessarily know whether he is in his office at all times. Clearly, a tradeoff exists between knowledge and privacy, and it will require further user testing to determine whether users prefer increased privacy or increased knowledge of the whereabouts of their fellow group members.

The monkey's inherent cuteness leads to several other possible issues. Because the monkey is so physically endearing, users may be more willing to overlook its potential for disruptiveness than they would be if we had used a gray metal box as an agent instead of a cute, stuffed monkey. It is difficult to conclude whether this is a positive or a negative feature. On one hand, we want end users to find the agent attractive enough that they will want to use it and interact with it. On the other hand, is it problematic if the agent is so physically appealing that users neglect to notice whether they actually find it to be useful? Can the monkey's cuteness actually override its potential utility?

An additional issue with the monkey's appearance is whether our system really exploits the affordances of having cute, furry monkeys as agents. Because the monkeys originated as stuffed animal puppets, and are therefore designed to be endearing and have a very huggable quality, many people instinctively feel the desire to touch them or pet them upon first seeing them. However, the current implementation of our system does not require any physical interaction between the users and the monkeys. Perhaps we might want to consider incorporating a tangible component, since the cute, fuzzy nature of the monkeys already evokes a natural inclination among people to physically interact with them. One suggestion would be to initiate audio chats by touching the monkeys, rather than by just leaning toward them. Perhaps squeezing a monkey's arm or patting its head could cause other remote monkeys to engage in a particular behavior. An interaction of this sort could be used as a tangible method of sending messages from one monkey to another, and would thereby utilize the monkey's touchable qualities to communicate with geographically distant monkey owners.

6.2 Future Work

We have many proposals for future versions of Monkey Business, including both ideas that our group discussed and planned to implement but did not have time or

resources for, as well as suggestions made by others for how the system could be modified and enhanced. Suggestions came from participants in our user studies, as well as from Media Lab sponsors and other students who had seen our demonstrations. Some of the future work ideas are small improvements that we could work on implementing in the near-term, while others are a bit more complex and would require a significant amount of time and dedication to execute.

One obvious improvement left for future work, as mentioned in Chapter 5, is to find and implement an alternative to multicasting streaming audio, since multicasting does not allow the audio packets to travel very far in a network. This would place fewer network restrictions on an environment in which Monkey Business could be fully functional.

A common proposal, as mentioned in the discussion section above, is to allow for greater end-user customization of the system. This could be realized in several ways. First of all, users could customize their monkeys to notify them only of certain types of events, or only of events in particular locations. For example, to re-visit our fictional users from the scenarios in the Introduction and Chapter 4, suppose that Susan has a really busy day ahead of her and does not want her monkey to notify her of anything going on elsewhere, except for activity in Mark's office, because she is collaborating with him on a project. She would be able to set her monkey to listen only for events from Mark's office, but not from anyone else's office. Or perhaps Susan is only interested in participating in audio chats with other system users, but does not want to be notified of their comings and goings through motion and proximity sensor event detection. In this case, she would be able to set her monkey to tell her when an audio chat was initiated, but not to tell her anything else. We envision this customization to be accomplished via an interface that would allow end users to specify which types of events and which locations they are interested in hearing about through their monkeys. Users would be able to change their customization settings at any time. They would also be able to save combinations of several settings as macros, so that they could just choose one pre-defined macro, instead of having to select several settings each time they use their monkeys.

Another type of customization would be to allow users to design their own sets of monkey behaviors to correspond with different events. For example, each user could

specify a combination of head, arm and body movements to map to each type of event that a monkey could detect. Similarly, users could choose the sounds that the monkey makes to indicate the locations in which events occurred. Currently, each monkey's sounds and movements are hard-coded into the system, and are therefore the same for every user, but this is the type of information that users should be able to personalize to suit their particular tastes.

Other future work possibilities include technical enhancements of the animatronics themselves. One obvious improvement would be to use Bluetooth as the communications protocol between each monkey and its computer, rather than a serial port connection. This would enable the monkey to be wireless, and would allow greater flexibility in setting up the monkey in a particular space, without any constraints of wires. It would also be more aesthetically appealing; currently the monkey has wires extending out of its body, which detracts from its physical appearance.

We might also be able to enhance the current audio component of Monkey Business. Right now, the system analyzes audio using amplitude as the discriminating factor by which it determines whether to play sound clips in other offices. However, with more sophisticated microphones and audio analysis techniques, we might be able to use the audio analysis to better recognize the specific activities that are occurring in each location. For example, perhaps we could use the audio analysis to recognize the sounds of a ringing phone, typing on a keyboard, or even certain words that are spoken in conversation. The monkey could then be used to represent specific activities, along the lines of what we tested in the user study. Because our audio analysis equipment and techniques are not sophisticated enough, we were not able to actually implement the detection of specific activities, although the design of the user study was based on the assumption that the system would work in this manner.

Another option for future work would be to add more servo motors inside the monkey, increasing its total number of degrees of freedom. This would add to its range of movements, and would allow for more interesting behaviors. Animating the monkey's face would be a particularly interesting enhancement, as it would allow the monkey to express emotions more readily. This could add another whole dimension to how the monkey is currently used, since individuals could then use the monkeys to express to

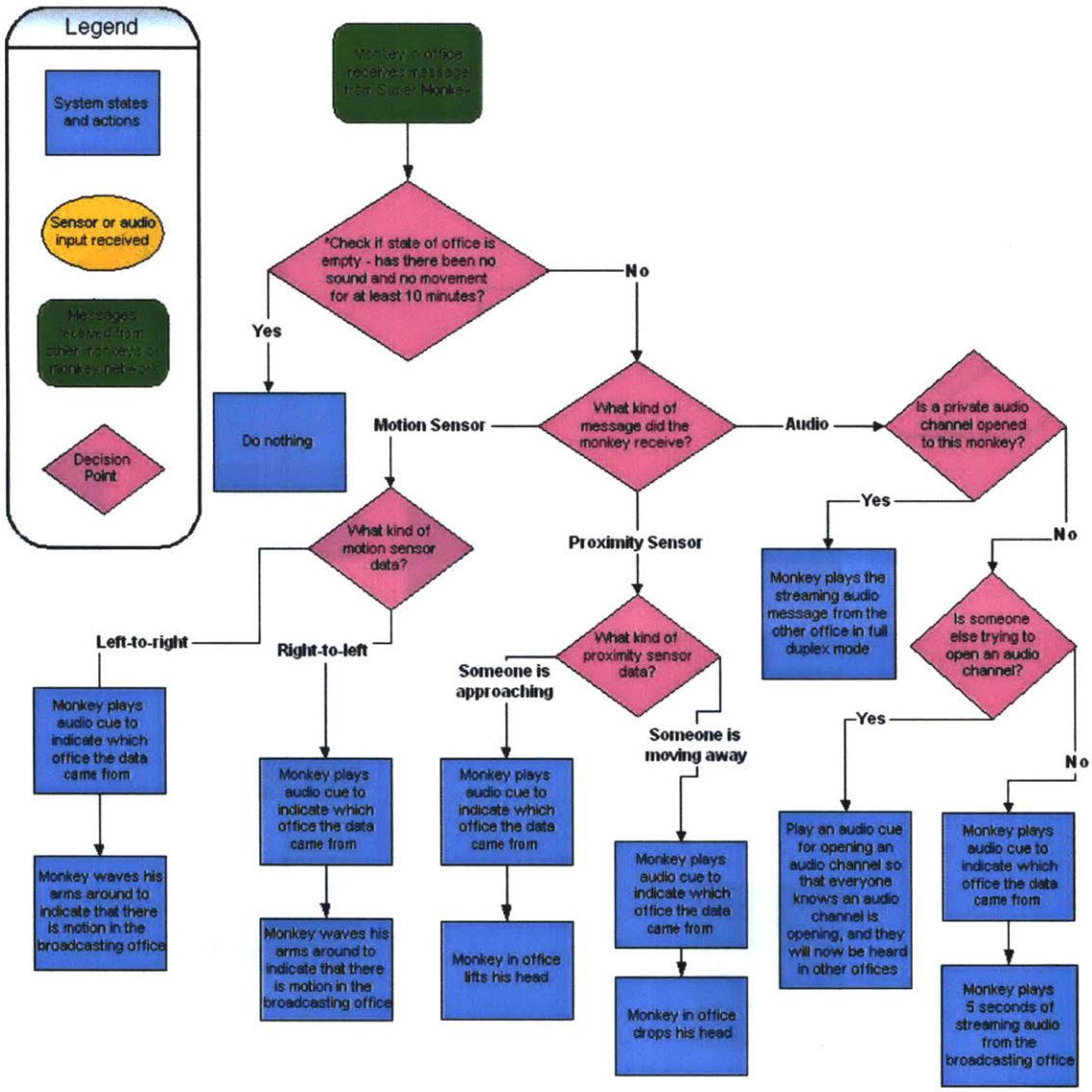
others how they are currently feeling, in addition to making others aware of their physical presence.

Finally, we could use animatronic agents other than monkeys as part of the system. End users would be able to choose their own mascots to act as their agents, and the diverse group of agents would communicate with each other, just as the monkeys do. This would probably call for changing the name of the system from Monkey Business, but it would allow for the greatest amount of end-user customization, as end users would not only select their agents, but would also specify how they wanted their agents to move and act upon receiving each type of event. This would be a challenge, as each agent would be able to move in different ways, so the representation of events would vary quite a bit from agent to agent. However, it would make the system much more personal for the end users, and would likely allow them to derive a fuller degree of joy from using it.

Appendix A

Monkey Business Flowcharts

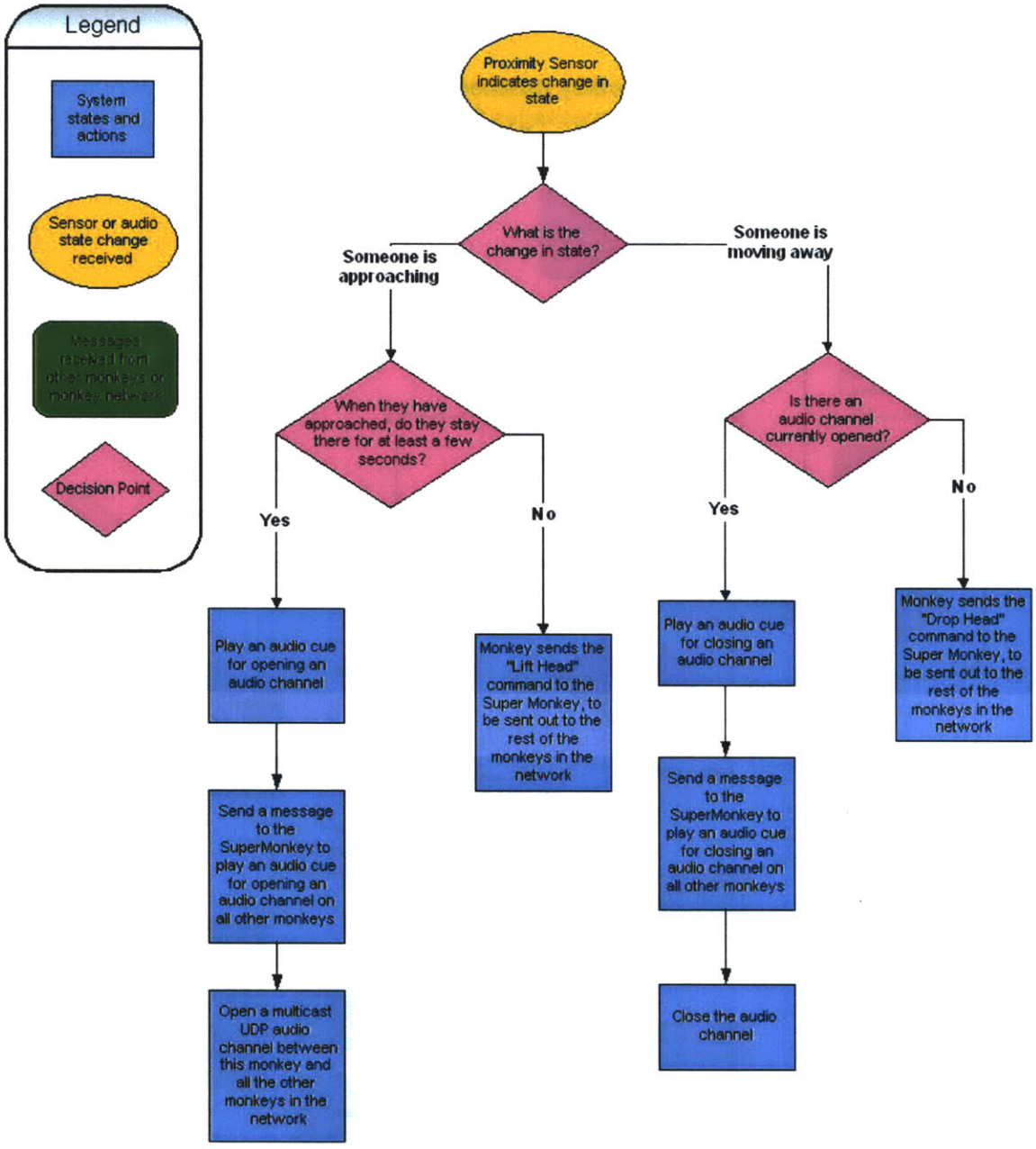
Monkey Business Flowchart 1:
Receiving messages from SuperMonkey, and deciding what to do with the information



These decisions are all made in the SuperMonkeySocket_DataArrival function, except for the Audio decisions, which are made separately.

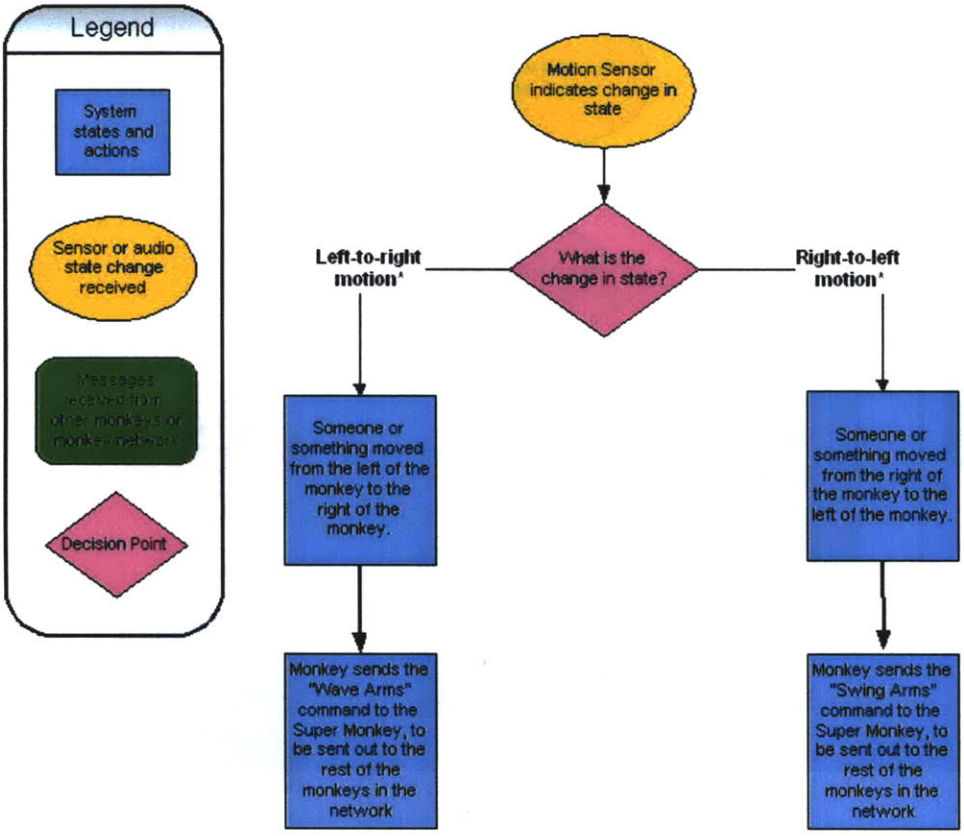
***Not yet implemented, right now it always assumes the office is not empty**

Monkey Business Flowchart 2:
Proximity Sensor Data and Sending Messages



These decisions are all made in the GetSensorData_Timer function.

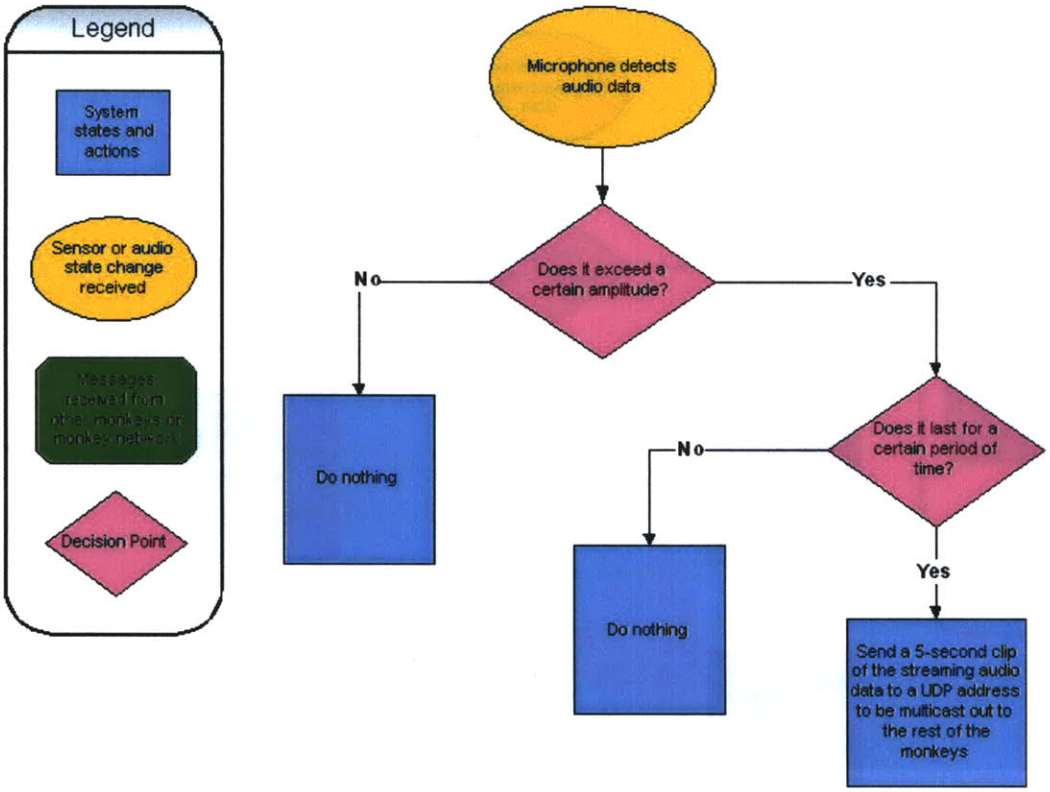
Monkey Business Flowchart 3:
Motion Sensor Data and Sending Messages



***Because there is not a meaningful difference between left-to-right motion and right-to-left motion, the monkey's behavior is very similar for both events. In both cases, the monkeys move their arms to indicate that motion was detected; the only difference is that in one case, Wave Arms, the monkey moves its arms in the same direction, and in the other case, Swing Arms, the monkey moves its arms in opposite directions. This was done mostly to have more variety in the monkey's movements.**

These decisions are all made in the GetSensorData_Timer function.

Monkey Business Flowchart 4:
Audio Data and Sending Messages



Appendix B

User Study Questionnaires

User Study Experience Sampling Method Survey Questions

1. What activity are you currently engaged in?

2. How many people, yourself included, are currently in the room?

3. On a scale from 1-7, how busy are you at the moment (7 = extremely busy and 1 = not at all busy)?

4. Did you notice if the monkey just recently moved?

5. If yes, how intrusive was the monkey on a scale from 1 to 7 (7 = very intrusive and 1 = not at all intrusive)?

6. What do you think the monkey was trying to demonstrate (7 = talking on phone, 6 = entering room, 5 = typing at keyboard, 4 = having conversation, 3 = leaving room, 2 = none of the above, 1 = I have no idea)?

7. If your answer to the previous question was "None of the above", what do you think the monkey was trying to demonstrate?

8. How straightforward (as in not ambiguous) was the monkey's demonstration (7 = I have no idea what the monkey was telling me and 1 = I know exactly what the monkey was telling me)?

9. Did you notice if there was any accompanying sound to the monkey's movement?

10. If yes, how intrusive was the sound on a scale from 1 to 7 (7 = very intrusive and 1 = not at all intrusive)?

11. What do you think the sound was trying to indicate?

12. Was the sound that the monkey made easy to interpret, i.e. did it have meaning to you (7 = very difficult to interpret and 1 = very easy to interpret)?

13. Did the sound make the expressions of the monkey clearer?

14. How intrusive was the monkey compared to yesterday (7 = much more intrusive and 1 = much less intrusive)?

15. Do you feel more accustomed to the monkey today than you did yesterday?

16. How urgent/important would you say the monkey's message seemed to be overall (7= very urgent and 1 = not at all urgent)?

User Study Follow-up Interview Questions

1. What expectations did you first have of the monkey before the study started?

2. Did you feel more or less comfortable with having the monkey in your office after a period of time?

3. a) If you did adjust to having the monkey in your office, how long did it take you to get accustomed to the monkey?
   b) If you were still aware of the monkey's novelty at the end of the study, how much longer do you think it would take before the novelty wore off?

4. How did the presence of other people affect your awareness and/or the intrusiveness of the monkey?

5. Did you notice whether the monkey's behavior changed from day to day. If so, what was different?

6. Was there a particular day when the monkey was most/least expressive and/or intrusive?

7. Given the movement limitations of the monkey, what kind of movements would you use?

8. When the monkey made sounds, did the sounds clarify what it was expressing? i.e. were they more helpful or just disturbing?

9. Did you notice that there were different kinds of sounds? If yes, what was the difference?

10. Were there any particular sounds that you found more/less informative?

11. If you were choosing the types of sounds that the monkey makes, what kind of sounds would you use?

12. Do you think the monkey should make "monkey sounds"?

13. Do you think the monkey is a good tool for intuitive expression in general? Why or why not?

14. In which situations do you think the monkey could be most helpful?

15. Did the monkey live up to your initial expectations?

Appendix C

Monkey Audio Code API

There are six classes of significance in this API: MA_Connection, MA_Manager, MA_Packet, MA_Source, MA_TcpConnection, and MA_UdpConnection. MA stands for Monkey Audio. Descriptions of each of these classes, as well as their public member functions, can be found in this appendix.

MA_Connection Class Reference

Audio pipe between **MA_Source** and arbitrary destination.

SUMMARY: Represents a single audio connection between local monkey audio source and an audio destination, such as a UDP socket; TCP socket; file; etc. Default class; specific implementations for different audio destinations.

USAGE: Call Connect() with appropriate parameters, to connect to audio destination. Call Subscribe() on **MA_Source** to subscribe this Connection to **MA_Source**; this will cause **MA_Source** to automatically supply Connection with recorded audio Use StartStreamingAudio() and StopStreamingAudio() to instruct Connection to handle or ignore recorded audio from **MA_Source** Supply ReceivedAudioCB, AudioStreamCB, and ReceivedConnectionCB, or use default functions, to define how to handle audio received from destination, or a connection request from destination Use Play() to supply data to the mixer. Play() takes a priority number, to help the mixer decide what to play and how to mix data.

Public Member Functions	
	MA_Connection ()
	Constructor.
	MA_Connection (**Observer** *manager)
virtual	**~MA_Connection** ()
	Destructor.
virtual bool	**connect** ()
	Connect to audio destination (UDP, TCP, file, etc).
virtual std::string	**getIP** ()
bool	**startStreamingAudio** (int byteCount)
	*Starts streaming audio on connection, from **MA_Source** to the destination of this connection.*
bool	**stopStreamingAudio** ()

	Stops streaming that was started by startStreamingAudio.
bool	**isStreaming** ()
	Indicates whether there is currently audio being streamed on this connection.
bool	**playClip** (const char *fName, **PlayPriority** pp)
	Plays a specified file on local sound device.
bool	**playBuffer** (const char *buffer, const int buflen, **PlayPriority** pp)
	Plays a specified audio buffer on local sound device.
bool	**retrieveAudio** (char *buffer, const int buflen, int *audioRetrieved)
	Retrieves a buffer of audio recorded on local sound device.
void	**setManager** (**Observer** *manager)
virtual void	**forwardPacket** (**MA_Packet** *pkt)
CString	**getType** ()

MA_Manager Class Reference

Object which maintains, creates, and destroys Connections and Source.

SUMMARY: Highest-level MA object; performs operations on **MA_Connection** and **MA_Source** objects. **MA_Manager** is primarily responsible for maintaining a list of current MA_Connections; providing global/static methods for creating new MA_Connections; and providing global/static methods for terminating connections and the **MA_Source**

USAGE: Static **MA_Manager** is automatically created. To make calls to the manager, claim the static object by using claimManager. After using the manager, release using releaseManager(*MA_Manager) Use the manager to create and destroy connection. Standard setup is:

```
MA_Manager* man = MA_Manager::claimManager();<BR>
man->createTCPConn(DEFAULT_TCP_PORT);<BR>
man->createUDPConn(DEFAULT_UDP_ADDRESS);<BR>
MA_Manager::releaseManager(man);<BR>
```

Connections can be destroyed individually, or you can just let the manager destroy them automatically when the program ends.

Public Member Functions	
void	**update** (std::string observableName, void *data, int dataLen)

	Called when a change has occurred in the state of the observable.
ConnectionID	**createUDPConn** (std::string groupIP)
ConnectionID	**createTCPConn** (int localPort)
MA_Connection *	**getConnection** (std::string IP)
MA_Connection *	**getConnection** (**ConnectionID** cID)
bool	**destroyUDPConn** (**ConnectionID** cID)
bool	**destroyTCPConn** (**ConnectionID** cID)

Static Public Member Functions	
static **MA_Manager ***	**claimManager** (int timeout=INFINITE)
	*Claims control over the **MA_Connection** manager The returned **MA_Manager** object can then be used to create, destroy, or access the MA_Connections.*
static void	**releaseManager** (**MA_Manager** *manager)

MA_Packet Class Reference

Data structure for data sent over MA network.

SUMMARY: Contains header information about data sent over the Monkey Audio network; also contains data itself.

USAGE: When receiving data, use the static **parsePacket()** method to create a **MA_Packet** object from a data string. This object can then be queried to retrieve data fields.

When sending data, use the static **createPacket()** method to package the necessary information into a **MA_Packet()**; then use **toString()** to create a character string that can be sent over the network.

PACKET STRING FORMAT:

The packets should have the following format:

```
"COMMAND;DESTIP;PP;DATALEN;DATA"
```

With the following sizes:

```
<char_8>;<char* variable>;<char_8>;<int_32>;<variable>
```

The DATALEN and DATA fields are interpreted differently, based on the COMMAND given. Here is a list of possible commands, and their corresponding interpretations of the other packet fields.

COMMAND = **DoNothing (0):**

- All fields are ignored.

COMMAND = **StartStream (1):**

- The DESTIP field contains the IP address of the connection to start streaming on.
- The PP field is ignored.
- The DATA field is ignored.
- The DATALEN field contains the number of milliseconds to stream audio; sending a value of 0 will enable continuous streaming.

COMMAND = **StopStream (2):**

- All fields are ignored.

COMMAND = **PlayFile (3):**

- The DESTIP field contains the IP address of the connection to play on.
- The PP field contains the PlayPriority ranking of the audio
- The DATA field contains the name of the file (including the path). File name should be null-terminated, if possible.
- The DATALEN field contains the size of the file name string, in characters.

COMMAND = **PlayBuffer (4):**

- The DESTIP field contains the IP address of the connection to play on.
- The PP field contains the PlayPriority ranking of the audio
- The DATA field contains the actual audio buffer to be played, in character-string form.
- The DATALEN field gives the length of the audio buffer, in bytes.

COMMAND = **RequestAudioToFile (5):**

- The DESTIP field is ignored.
- The PP field is ignored.
- The DATA field contains the name of the file to create, containing the requested audio. This file MUST be null-terminated (be followed by at least one 0-valued character).
- The DATALEN field contains the number of bytes to write to the file. It does NOT specify the length of the file name!!!

COMMAND = **AudioRequestedInFile (6):**
This command is sent *from* the MonkeyAudio code, to the process that requested audio.

- The DESTIP field is ignored.
- The PP field is ignored.
- The DATA field confirms the name of the file; this will be a null-terminated string.
- The DATALEN field contains the number of bytes actually recorded to the file; this may be different than the number of bytes requested!

COMMAND = **CreateUDPconnection (7):**

- The DESTIP field is ignored.
- The PP field is ignored.
- The DATA field contains the IP address of the UDP group (eg. "230.255.1.9").
- The DATALEN field contains the length of the IP address of the UDP group.

COMMAND = **TerminateUDPconnection (8):**

- The DESTIP field is ignored.
- The PP field is ignored.
- The DATA field contains the IP address of the UDP group (eg. "230.255.1.9").
- The DATALEN field contains the length of the IP address of the UDP group.

COMMAND = **TerminateConnection (9):**

- All fields are ignored. This connection is immediately broken.

Public Member Functions	
MA_Command	**getCommand** ()
	Retrieves the MA_Command field from a packet.
void	**setCommand** (**MA_Command** command)
	Alters the MA_Command field in a packet.
char *	**getDestIP** ()
	Retreives the destination IP address field from a packet.
void	**setDestIP** (const char *destIP)
	Sets the destination IP address field in the packet.
int	**getDataLength** ()
	Retrieves the length of the data field in the packet.
char *	**getData** ()
	Retrieves the data/audio field from the packet.
void	**setData** (const char *data, const int dataLength)

	Sets the data field of the packet.
PlayPriority	**getPP** ()
	Retrieves the PlayPriority of the data field Only meaningful if data field contains audio data.
void	**setPP** (**PlayPriority** pp)
	Sets the PlayPriority value of the data Only meaningful if data field contains audio data.
char *	**toString** ()
	*Creates a string representation of the **MA_Packet** object, which can be sent over a socket to another monkey.*
int	**toStringLength** ()
	*Returns the length of the string which would be returned by **toString()**.*
	~MA_Packet ()
	Destructor.

Static Public Member Functions	
static **MA_Packet** *	**parsePacket** (char *recvdData)
	Creates a packet object from a received data string.
static **MA_Packet** *	**createPacket** (**MA_Command** command, const char *destIP, **PlayPriority** PP, int dataLength, char *data)
	*Creates a packet from user-defined packet fields Used for creating a packet from local data or audio, which can then be sent over the network in string form by using **toString()**.*

MA_Source Class Reference

Provides static access to sound card.

SUMMARY: Observable wrapper of sound card functionality. Maintains a circular buffer of recently recorded buffers, as well as a mixer with buffers to be played. Always recording and, if there is data available to play, always playing.

USAGE: Call Initialize() with a sound card device number to set up. This can only be called once; additional calls will be ignored. Use Subscribe() to enable a **MA_Connection** to retrieve audio data in real time from the microphone. Use RequestAudio() to retreive audio data recently recorded. Use Play() to supply data to the

mixer. Play() takes a priority number, to help the mixer decide what to play and how to mix data.

Public Member Functions	
void	**startDevice** (int deviceNum=WAVE_MAPPER)
	*Sets up **MA_Source**; call this in place of constructor (which is private).*
void	**stopDevice** ()
	Stops the sound device - terminates record/play thread.
bool	**isInitialized** ()
	*Reports whether or not the **MA_Source** is initialized, through **startDevice()**.*
int	**getAudioDevice** ()
	Generates a list of audio devices, prints list to console.
bool	**playFile** (const char *name, **PlayPriority** pp)
	Wrapper for wavAudio playFile function.
bool	**playAudio** (const char *buffer, const int buflen, **PlayPriority** pp)
	Wrapper for wavAudio playBuffer function.
int	**requestAudio** (const int buflen, char *buffer)
	Copies and returns a chunk of audio from the circular record buffer.
void	**haltPlaying** ()
	Kills any files being played on audio device.
void	**notifyObservers** ()
	*Overrides default function in abstract class **Observable**.*
void	**setWaveFormat** (WAVEFORMATEX *newFormat)
	Defines the format of audio on the device.
WAVEFORMATEX *	**getWaveFormat** ()
	Returns the format of audio being used on the device.
DWORD	**monitorRecordStream** ()
	Threaded function that monitors for recorded audio data.
DWORD	**monitorPlayStream** ()
	Threaded function that manages incoming audio data.

Static Public Member Functions	
static **MA_Source** *	**claimSource** (int timeout=INFINITE)
	Claims control over the source object (the sound card).
static void	**releaseSource** (**MA_Source** *src)
	*Returns the source claimed by **claimSource()**.*
static bool	**saveWaveFile** (char *fname, char *data, int len, WAVEFORMATEX *waveformat)

MA_TcpConnection Class Reference

Audio pipe between **MA_Source** and TCP socket.

SUMMARY: Represents a single audio conection between local monkey audio source and a TCP socket

USAGE: Call Connect() with appropriate parameters, to connect to audio destination. Call Subscribe() on **MA_Source** to subscribe this Connection to **MA_Source**; this will cause **MA_Source** to automatically supply Connection with recorded audio Use StartStreamingAudio() and StopStreamingAudio() to instruct Connection to handle or ignore recorded audio from **MA_Source** Supply ReceivedAudioCB, AudioStreamCB, and ReceivedConnectionCB, or use default functions, to define how to handle audio received from destination, or a connection request from destination Use Play() to supply data to the mixer. Play() takes a priority number, to help the mixer decide what to play and how to mix data.

Public Member Functions	
	MA_TcpConnection (int localPort)
	Constructor, requires local TCP port as argument.
virtual	**~MA_TcpConnection** ()
	Destructor.
bool	**connect** ()
	Connects to a default TCP port and IP address.
bool	**connect** (const std::string &foreignAddress, const unsigned short foreignPort)
	Connects to a specified TCP port and IP address.
bool	**listenForConnection** ()
	Sets the TCP socket to accept a connection on local IP/port.
int	**handleSourceAudio** (const char *buffer, const int len)

	Handle incoming audio from local sound device.
int	**handleDestAudio** (const char *buffer, const int len)
	Handle incoming data from TCP socket.
void	**update** (std::string obsName, void *data, int dataLen)
	*Called by **Observable** object when there is new data.*
std::string	**getIP** ()
void	**forwardPacket** (**MA_Packet** *pkt)

MA_UdpConnection Class Reference

Audio pipe between **MA_Source** and UDP socket.

SUMMARY: Represents a single audio conection between local monkey audio source and a UDP socket

USAGE: Call Connect() with appropriate parameters, to connect to audio destination. Call Subscribe() on **MA_Source** to subscribe this Connection to **MA_Source**; this will cause **MA_Source** to automatically supply Connection with recorded audio Use StartStreamingAudio() and StopStreamingAudio() to instruct Connection to handle or ignore recorded audio from **MA_Source** Supply ReceivedAudioCB, AudioStreamCB, and ReceivedConnectionCB, or use default functions, to define how to handle audio received from destination, or a connection request from destination Use Play() to supply data to the mixer. Play() takes a priority number, to help the mixer decide what to play and how to mix data.

Public Member Functions	
	MA_UdpConnection (int localPort)
	Constructor, requires local UDP port as argument.
virtual	**~MA_UdpConnection** ()
	Destructor.
bool	**connect** ()
	Connects to a default UDP group address.
bool	**connect** (const std::string &groupIP)
	Connects to a specified UDP group address.
int	**handleSourceAudio** (const char *buffer, const int len)
	Handle incoming audio from local sound device.
int	**handleDestAudio** (const char *buffer, const int len)

	Handle incoming audio from UDP socket.
void	**update** (std::string obsName, void *data, int dataLen)
	*Called by **Observable** object when there is new data.*
void	**setMulticastTTL** (const int multicastTTL)
	Sets "Time To Live" value for outgoing packets.
std::string	**getIP** ()
void	**forwardPacket** (**MA_Packet** *pkt)

References

[1] Bly, S. A., Harrison, S.R., Irwin, S. (1993). Media Spaces: Bringing People Together in a Video, Audio and Computing Environment. Communications of the ACM. 36 (1) pp. 28-47.

[2] Cassell, J., Bickmore, T., Vilhjalmsson, H., Yan, H. (2000). More Than Just A Pretty Face: Affordances Of Embodiment. Proceedings of the 5th Int'l Conference On Intelligent User Interfaces, pp. 52-59.

[3] Dautenhahn, K., Ogden, B., Quick, T. (2002). From embodied to socially embedded agents – implications for interaction-aware robots. Cognitive Systems Research 3 (3) pp. 397-428.

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

[5] Eggen, B., Rozendaal, M. and Schimmel, O. (2003). Home Radio - Extending the Home Experience beyond the Boundaries of the Physical House. Proceedings of the HOIT 2003 (Home Oriented Informatics and Telematics), University of California, Irvine.

[6] Fish, R. S., Kraut, R. E., Root, R. W., and Rice, R. E. (1992). Evaluating Video as a Technology for Informal Communication. Proceedings of the ACM Conference on Human Factors in Computing Systems CHI '92, 37-48, ACM Press.

[7] Fogarty, J., Hudson, S., Atkeson, C., Avrahami, D., Forlizzi, J., Kiesler, S., Lee, J. and Yang, J. (2005) Predicting Human Interruptibility with Sensors. ACM Transactions on Computer-Human Interaction (TOCHI). 12 (1) pp. 119-146.

[8] Greenberg, S. (2004). Collaborative Physical User Interfaces. Report 2004-740-05, Department of Computer Science, University of Calgary, Alberta, Canada.

[9] Gross, T. (2003). Ambient Interfaces: Design Challenges and Recommendations. Proceedings of the 10th International Conference on Human-Computer Interaction, pp. 68-72.

[10] Hindus, D., Ackerman, M., Mainwaring, S., and Starr, B. (1996). Thunderwire: A Field Study of an Audio-Only Media Space. Proceedings of the Conference of Computer Supported Co-operative Work, pp. 238-247.

[11] Isaacs, E., Walendowski, A., and Ranganthan, D. (2002). Hubbub: A sound-enhanced mobile instant messenger that supports awareness and opportunistic interactions. Proceedings of the Conference on Computer Human Interaction, pp. 179-186.

[12] Ishii, H., Ullmer, B. (1997). Tangible Bits: Towards Seamless Interfaces Between People, Bits and Atoms. ACM CHI '97, 234-241, ACM Press.

[13] Kraut, R., Fish, R., Root, R., and Chalfonte, B. (1993). Informal Communication In Organizations: Form, Function, and Technology. In Baecker, R. (ed.), Groupware and Computer-Supported Co-operative Work, Morgan Kaufmann, pp. 287-314.

[14] Lakshmipathy, Vidya. SimPhony: voice group communication. MIT Masters thesis, February 2004.

[15] Larson, R. and Csikszentmihalyi, M. (1983). The Experience Sampling Method. In H. T. Reis (Ed.), Naturalistic Approaches to Studying Social Interaction: New Directions for Methodology of Social and Behavioral Science. San Francisco, CA: Jossey-Bass.

[16] Lock, S., Allanson, J., Phillips, P. (2000). User-Driven Design of a Tangible Awareness Landscape. Proceedings of Designing Interactive Systems '00: Processes, Practices, Methods, & Techniques 2000. pp. 434-440.

[17] Mackay, W. E. (1999). Media Spaces: Environments for Informal Multimedia Interaction, in Beaudouin-Lafon, M. (ed.), Computer Supported Cooperative Work, John Wiley & Sons, pp. 55-82.

[18] Marti, S., Schmandt, C. (2005). Physical Embodiments for Mobile Communication Agents. Proceedings of the 18th annual ACM symposium on User Interface Software and Technology. pp. 231-240.

[19] Marti, Stefan J. W. Autonomous Interactive Intermediaries: Social Intelligence for Mobile Communication Agents. MIT PhD thesis, May 2005.

[20] Pederson, E. R., and Sokoler, T. (1997). AROMA: abstract representation of presence supporting mutual awareness. Proceedings CHI '97, ACM Press, pp. 51-58.

[21] Vallejo Rosas, Gerardo M. ListenIN: Ambient Auditory Awareness at Remote Places. MIT Masters thesis, September 2003.

[22] Whittaker, S., Frohlick, D., and Daly-Jones, O. (1994) Informal Workplace Communication: What Is It Like And How Might We Support It? Proceedings of Human Factors in Computing Systems, pp. 131-137.