tunA: Shared Audio Experience

Julian Moore
Advanced Scholar / AIB Fellow
University of Limerick / Media Lab Europe (2004)

Bachelor of Computer Engineering with First Class Honours
University of Limerick (2002)

Submitted to the College of Informatics & Electronics in fulfilment of the requirements for the degree of

Master of Science
at the
UNIVERSITY OF LIMERICK

Timothy Hall, University of Limerick
Stefan Agamanolis, Media Lab Europe

Submitted to the University of Limerick, September 2004

Keywords: Mobile Audio, MP3, Ad-Hoc Networking, Synchronisation, Instant Messaging, Social Networks, P2P, Peer to Peer, Shared Experience, Distributed Systems, tunA, Proximity Networks, Wireless Devices, Personal Music Players

Abstract

This thesis presents *tunA* – a 'socialising MP3 player'. An evolution of the walkman, it is a peer-to-peer software application for mobile devices that as well as functioning as a normal MP3 player, allows users to locate nearby peers, view their profiles, send instant messages, and wirelessly share music. Rather than simply exchanging music libraries, the software streams audio from device to device in a tightly synchronised fashion, so that all parties hear the same audio, at the same time. This combination of discovery mechanism, profile and message swapping, and 'shared audio experience' provide a platform for people to encounter and engage with others of similar tastes and interests. A comprehensive logging mechanism completes the application, and extends it into a valuable tool for the study of social dynamics.

Technically, *tunA* sports a number of unique characteristics, several of which are the subject of a provisional United States patent. These features include: a custom algorithm for synchronising the playback of streaming media in ad-hoc networks, a proximity-based instant messenger, the ability to 'bookmark' favourite peers / audio, and a novel user interface (including a bespoke input mechanism for touch screen mobile devices).

Ethnographically, *tunA* aims to uncover and explore new forms of social interaction, evolution of musical tastes, audio portraits, listening habits, and music distribution.

Declaration

This thesis is presented in fulfilment of the requirements for the degree of Master of Science at the University of Limerick.

It is entirely my own work and has not been submitted to any other University or higher education institution, or for any other academic award in this University. Where use has been made of the work of others it has been fully acknowledged and referenced.

_____

Julian Moore, September 2004

For my little ‘wonder geek’ Danica, and all her love, support, and pusas

Acknowledgements

I would like to take this opportunity to thank the many individuals who assisted me throughout the course of this work. I owe a great debt of gratitude to Arianna Bassoli, my partner in the *tunA* project and master of *tunA's* ethnographic dimension. I must also praise my supervisors Timothy Hall, and Stefan Agamanolis for their generous advice and direction. Thanks to Nick Hawes for his T9 reference code. Thanks also to my friends in the Human Connectedness group, and the *eighteen* flatmates who have filled my time at the lab with laughter. Finally, I would like to thank my parents for my rugged good looks, and their boundless encouragement and support.

This work was made possible by AIB, and the other partners of Media Lab Europe, the European research partner of the MIT Media Lab.

List of Figures

List of Tables

Contents


TUNA

1 Introduction

1.1 Motivation

In the thirteenth century Frederick II, the then Roman Emperor and King of Southern Italy began a series of dubious experiments on babies to determine if humans possessed an innate knowledge of languages such as Hebrew, Greek, Latin, and Arabic (Agamanolis, 2004). To this end he ordered that a group of infants be raised who would never hear speech. To ensure this, the foster mothers who cared for the babies were given strict instructions to feed and bathe the children, but to never speak to, or handle them. All the infants died; as the Italian historian Salimben de Parma put it:

> *"…children could not live without clapping of the hands, and gestures, and gladness of countenance, and blandishments."*

Like many of humanities mistakes, this was to be repeated. By the 1940's new theories on the spread of disease through germs had taken root in western medicine, and were greatly influencing the treatment of orphaned children (Stien & Kendall, 2004). Institutional babies were kept fed, clothed, warm, and clean, but to lessen their exposure to germs, were not played with, handled, or held. Much to the bewilderment of the doctors, the children became withdrawn, sickly, and underweight. In fact, their resistance to disease actually decreased, with some 40% of the children who caught measles dying, compared to a mortality rate of only 0.5% in the general population.

The point of these gruesome stories is that interaction with others is a very real, biological human need. We are by nature 'open-loop' systems; we depend on social contact with others to help control our physiology, and without this input are prone to depression, illness and disease (Lewis, Amini et al., 2003). Medically this phenomenon of is known as 'limbic regulation' and it effects almost every aspect of our lives. It is for instance, the reason why the menstrual cycles of women who live together, align over time. It has even been suggested that isolation constitutes a risk to our health as tangible as that of smoking, obesity, or lack of exercise (House, Landis et al., 1998).

Despite these documented risks, we are becoming an increasing private people on both a societal, and an individual level. Drawing on nearly 500,000 interviews over the last quarter century (Putnam, 2000) warns that our stock of 'social capital' is plummeting. We sign less petitions, belong to fewer organisations, are less likely to vote, are becoming increasingly detached from our neighbours, and even socialize less with our families. Paradoxically, the very modern technology that shrank the world to a 'global village' is distancing us from our peers. Work demands that many of us travel or live far apart from our families; distances that are made even greater by the advent of cheap air travel. Television and the Internet are providing poor domestic surrogates for more group-oriented activities, and a pasty combination of mobile phones and credit cards is increasingly becoming our interface to the outside world.

One of the developments to expedite these trends, was the arrival of the first Sony Walkman (Du Gay, 1997). With its release at the end of the seventies, it ushered in a new era of 'personal technologies', little machines that we religiously carry on (and eventually someday in?) our bodies. With their help, people can surf the Net in parks, listen to music on trains, watch movies on airplanes, and play videogames on buses - almost never interacting with

strangers. GameBoys, PDAs[1], mobile phones, laptop computers, MP3 players - each successive device adds another layer to the technological bubble that disconnects us from our environment (Sartwell, 1999).

However, a backlash against this behaviour is emerging. In explaining the recent 'toothing' phenomenon (Terdiman, 2004) for example, reveals how:

> *'…strangers on trains and buses and at bars and concerts hook up for clandestine sex by text messaging each other with their Bluetooth-enabled cell phones or PDAs…'*

tunA's raison d'être is to fan the flames of this rebellion. An evolution of the walkman, it is a peer-to-peer software application for mobile devices that as well as functioning as a normal MP3 player, allows users to locate nearby peers, view their profiles, send instant messages, and wirelessly share music. Rather than simply exchanging music libraries, the software streams audio from device to device in a tightly synchronised fashion, so that all parties hear the same audio, at the same time. This combination of discovery mechanism, profile and message swapping, and 'shared audio experience' provide a platform for people to encounter and engage with others of similar tastes and interests. A comprehensive logging mechanism completes the application, and extends it into a valuable tool for the study of social dynamics.

1.2 Example Scenarios

As is true of many Media Lab projects, *tunA* is a multidisciplinary effort, comprising a technical solution to a social research problem. The following scenarios are intended to demonstrate the need for, and use of, a *tunA*-based device. While doing so they hopefully lend some context to the discussion which follows. All characters and events are fictitious.

[1] PDA = Personal Digital Assistant

1.2.1 Serendipitous Exchanges

Alice and Bob grew up in the same suburb, attended adjacent schools, and rode the same bus together every morning for six years, without ever speaking a word to each other. They moved in separate social circles, and while vaguely aware of each other, had little enough cause to approach one another. That changed years later in a dingy little bar in a small town outside Paris, where they met by chance and discovered they had many similar tastes, in music, literature, films, and politics. They have remained fast friends since.

By mining the environment for other users, *tunA* acts as a catalyst for chance encounters between people with similar interests. Furthermore by immersing listeners in a shared audio experience it provides a common medium for fostering new social bonds. If the couple in this scenario had both used *tunA*-based walkmans on their rides to school; enjoying the same music, cheering at the same sports results, or laughing at the same comic routines, they may have discovered each other years earlier. It is precisely these serendipitous exchanges that *tunA* is designed to encourage and support.

1.2.2 Viral Distribution

Hank is twenty-eight going on forty. The soulless commutes to his firm's offices in the city depress him, as does the 'Quit Smoking for Dummies' audio book he's half-listening to. You have to read something on the train or the girl opposite you will think you're staring at her no matter where you try to look. In New York everyone's afraid of strangers, and not wanting a reputation as the 'Q train psycho', Hank listlessly pans the Wall Street Journal, or stares at the ads on the ceiling. They're all for dating services.

Enough! He crumples the paper in his lap, pinches his temples as though to squeeze the boredom from his head, and looks up. Standing in front of him is a beautiful Hispanic woman, who like Hank is wearing a tunA-enabled walkman.

Switching to the 'nearby users' mode he spots her avatar - 'Carolina'. Hank clicks the 'tune in' button and a few seconds later, he's listening to the same audio stream as her. It's a track from Linkin Park's latest album (Carolina likes alternative rock). After a couple of stops she steps off the train, and Hanks walkman falls silent. He liked the song though, so he 'bookmarks' it. Later, at the office, he cradles his tunA device and it automatically downloads the entire album from his online subscription service.

Other people have more influence on us than we would like to believe. They can change the way we think, act, and feel. Our own tastes are a complex function of our interactions with those of others; a sobering fact that marketing departments have known for years. Everyday swathes of new customers are won over to a brand of deodorant/shaving gel/jocks on the basis that David Beckham or Tiger Woods consume them. When launching the original walkman, Sony hired attractive young men and women to wander the streets and parks of Japan, listening to music and offering others the chance to try them out (Sony, 2004), and in the example above Hank's musical tastes evolved through his exposure to Carolina's.

Applications like *tunA* have the potential to invert the current channels of music distribution[2]. Instead of the top-down model of a cartel of large record labels manipulating the charts (and consequently popular choice) with inflated sales figures and inoffensive, formulaic rock bands, a more grassroots system of peer-to-peer recommendations may evolve. We can already see the beginnings of this with sites like *Amazom.com* recommending items by comparing the purchase histories of multiple shoppers. The exploration of these possibilities is one of the ultimate goals of this line of research.

[2] Unless the major record labels follow Sony, and hire thousands of attractive people to ride the subways listening to their content of course…

1.3 Contributions of This Work

Technically, *tunA* sports a number of unique characteristics, several of which are the subject of a provisional United States patent (see Appendix C). These features include: a custom algorithm for synchronising the playback of streaming media in ad-hoc networks, a proximity-based instant messenger, the ability to 'bookmark' favourite peers and audio, and a novel user interface, including a bespoke input mechanism for touch screen mobile devices.

Ethnographically, *tunA* aims to uncover and explore new forms of social interaction, and music distribution. To facilitate this, the software records all internal and external events for later review by a social studies researcher. Typical research questions in this domain include:

- **Listening habits** – Is the knowledge that others may be aware of what one is listening to, sufficient to alter to a person's choice of music? To what extent, and what can be done to mitigate this?
- **Evolution of tastes** – How do musical tastes evolve? Do the musical preferences of our peers exert a significant influence on our own?
- **Audio portraits** – By cross-referencing location data with a record of the music an individual has been exposed to over time, would it be possible to build up an 'audible portrait' of an area, and how would this be best visualised / embodied?
- **Music distribution** – What might the impacts be on music distribution of a system of recommendation like this? Would the overall popularity of certain genres or artists be affected? How?

The purview of this thesis is limited to the technical aspects of *tunA*, the innards of which are fully exposed in Chapter Three. However, a limited discussion of several social studies is presented in Chapter Four.

1.4 Thesis Overview

This thesis opens with a blend of jarring anecdotes, and medical fact, as a synopsis of the motivation behind the Human Connectedness group at Media Lab Europe, under whose auspices this work was conducted. *tunA* and its societal rationale are both introduced through a number of example scenarios, and the technical and ethnographic contributions of this project are stated. The remainder of this document unfolds as follows:

Chapter Two presents a literary review of related research in this area, divided into three main categories: Mobile Audio, Social Networks, and Shared Experience. Chapter Three describes the design and implementation of *tunA*, broken down by core functionality: user interface, peer discovery, media playback and streaming, audio synchronisation, instant messaging and database logging. Chapter Four summarizes the research methodologies and results of a number of user studies, as well as outlining the earlier work on which these were based. It also provides details of the other live exhibitions and installations of *tunA*. Finally, a selection of technical and ethnographic conclusions, and several vectors for future work are offered in Chapter Five. These include a custom hardware platform, various extensions to the existing code base, and some early plans for licensing and distributing this technology.

2 Background / Related Work

2.1 Overview

To provide some context for the evaluation of the claims made in the introduction as to the ingenuity, imperative, and efficacy of *tunA*, this chapter presents a literary review of related research in three main areas: Mobile Audio, Social Networks, and Shared Experience, or in other words, the what, why, and how of this work.

The first section traces the technical genealogy of mobile audio devices like *tunA*, from the humble beginnings of the Sony walkman, through to the feature-rich, Internet-aware MP3 players of today. Next, Social Networks are introduced and examined, beginning with 'online community' websites like *Friendster.com* and moving onto viral message exchange, and proximity networks. Lastly the notion of shared activity as a means of social bonding is explored with examples drawn from the worlds of sport, music, and television.

Throughout the review key points are identified and summarised in the left-hand column and from these a series of design goals for this domain are derived and explained in the final section. These goals form the basis of an iterative process of development, fully documented in the third chapter.

2.2 Mobile Audio

This section defines what *tunA* is in terms of other mobile devices. The original walkman forms a basic frame of reference, and from there the discussion moves to synchronised audio streams, shared audio spaces, and the environment as a musical interface. The projects are presented in order of increasing relevance, and the segment concludes with a critique of recent work (*Awaire*, *Bubbles*, *Sound Pryer*) that is similar in scope and design to *tunA*.

2.2.1 Walkman, Discman, MiniDisc – Sony, Japan

Figure 2-1 *An early prototype of the Sony walkman*

Launched by Sony in 1979, the TPS-L was the world's first walkman. Lacking any facility for recording, and initially lost in a quagmire of scepticism, this humble cassette player rocketed from obscurity to ubiquity in a matter of years.

The last decade has seen a shift from analogue to digital technology, with CD's, MiniDiscs, flash memory, and small, robust hard drives replacing tape as a storage medium for all but its most ardent fans. Other than these changes and the improvement in fidelity they have rung in, the overall design and functionality of the walkman has largely stagnated for over two decades. Development work has focused on improving battery life (now over forty hours per charge), storage (up to 60GB of capacity – the equivalent of roughly 15,000 tracks) and fidelity (noise-cancelling headphones give almost perfect reproduction even in boisterous environments).

Figure 2-2 *The iPod from Apple*

The common thread here is that none of these modifications have challenged the tradition of the walkman being an inherently solitary device. The next projects – do exactly that.

2.2.2 MyFi, TuneCast – Neuros & Belkin, USA

Figure 2-3 *Neuros MyFi MP3 player / FM transmitter*

Synchronous
The human ear will assume two audio signals are 'coherent' (i.e. from the same source) if they arrive within 30ms of each other. This is the approximate level of accuracy required to allow multiple users to 'bop along' to the same piece of music.

Figure 2-4 *Belkin TuneCast dongle*

The Neuros *MyFi* is a commercially available MP3 player with a built in FM tuner (Neuros, 2004). Its firmware allows it to stream audio from one device to another, or indeed to any other FM compatible radio in range. It does this in the form of a traditional analogue broadcast. The Belkin *TuneCast* is a small piece of hardware that attaches to the headphone socket of an iPod to offer similar functionality. Only the products form factor tailors it to the iPod, it will accept input from any audio source of comparable impedance.

These devices have a similar target audience to *tunA*; even the name 'MyFi' is suggestive of a personal radio station. The obvious advantage to using a cheap FM module to transmit audio is that the high speed of radio waves (300,000,000 m/s in free space) means that there is no perceivable lag between the audio on multiple peers, giving tightly synchronised audio across multiple peers. The convoluted radio stacks, and unpredictable operating system overheads involved in more complicated systems make this behaviour hard to replicate. Both devices also share the same weakness - by relying on simple analogue transmissions they are prevented from offering more advanced features such as peer discovery, shared play lists, or instant messaging. They rely on 'out of band signalling' to provide all this - i.e. users have to verbally agree on a common frequency to transmit/receive on. This reduces an otherwise interesting feature to little more than a convenient method for wire-free playback on a car stereo. In addition, the effective transmission range in each case is clipped to less than 20 feet.

2.2.3 Sonic City – Viktoria Institute, Sweden

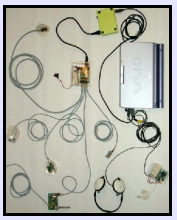


Figure 2-5 *Sonic City's hardware platform*

Engaging
Personal devices should not isolate their users from their surrounding environments as most existing units do, but instead should encourage or at least allow engagement with the environment and populace surrounding them.

Sonic City is a context-aware mobile audio player that shapes the music being played according to the input from a range of environmental and biometric sensors (Gaye, Maze et al., 2003). Light, sound, metal, acceleration, and galvanic skin response sensors are all wired to a microcontroller which sends this data to a laptop via a MIDI[3] stream. Once there, it is used as the input to various functions which apply tempo, frequency, and delay effects to the audio stream.

While somewhat orthogonal to the other projects presented here, Sonic City is noteworthy for using the immediate environment as musical interface. Whereas the walkman was designed to immerse the user in a world of sound, this has as its primary motive the reverse intention. It is in some sense then, an 'anti-walkman', albeit one that is still very much in the development stages, as evidenced by the mess of cables in figure 2-5. More critically, Gaye et al. have yet to conduct any studies to determine the value or effect of this experience.

2.2.4 Sotto Voce – Xerox PARC, USA

Figure 2-6 *Sotto Voce guidebook*

Sotto Voce is an electronic guidebook developed at Xerox PARC for browsing museums (Aoki, Grinter et al., 2002). Running on WiFi enabled iPaqs[4]; the software consists of a visual interface from which users can select an audio description of any exhibit. Significantly, users of the device are 'paired' such that respective partners can hear the same audio cues as each other, for a 'shared audio experience'.

[3] MIDI – Musical Instrument Digital Interface, protocol to connect computers/musical devices

[4] iPaq – Line of Pocket PC's from HP, now synonymous with Windows Mobile

More specifically for two paired visitors A and B, when visitor A selects an object on their device, they always hear their own audio clip. If A is not currently playing an audio clip, but her companion B is, then B's audio clip can be heard on A's device. In other words, audio clips are never mixed, and A's device always plays a personal clip (selected by A) in preference to an eavesdropped clip (selected by B). During field trials Aoki et al. observed marked differences between participants when using the audio augmented devices as opposed to the traditional paper guides. Conversations for example, tended to be extended multi-turn exchanges instead of basic stimulus-response dialogs. In addition, there was a greater relative movement between partners, (attributed to the shared audio making them feel connected over greater distances), and often eavesdropped content audio from another device served to pique interest and prompt re-engagement between partners.

Sotto Voce is not without its share of deficits. Technically the system is a 'hack' in that no waveform data is streamed from one device to another, instead a series of simple 'start playing clip 10', 'stop playing clip 7' style commands are swapped between clients to create the illusion of synchronised audio. As a consequence, every device must have an identical local store of audio, so the approach cannot be extended to a more general case, such as individual music libraries. Furthermore, the *Sotto Voce* trials took place in a highly constrained environment, between people who already had an existing social bond. As such there is no element of peer discovery or serendipitous encounter, and presumably each participant already possesses several commonalities.

2.2.5 Pocketster – Simeda, Hungary

Figure 2-7 *Pocketster C# app from Simeda*

Personal *Anonymous file swapping, while it may in some cases be a desirable feature is a poor substitute for personal contact; the fan is more important than the music.*

Pocketster is a recent software application from Romanian outfit Simeda (Orlowski, 2004). A variation on the existing Apple Rendezvous stack (Apple, 2004b) it is essentially a web server that advertises its presence to neighbouring devices, and is closely related to HocMan (see section 2.3.4). By hosting a web page that contains links to audio files, or alternatively a flash audio control (Wimpy, 2004), users music collections can be streamed or downloaded by others.

While Rendezvous is an elegant way to advertise web services, this is a somewhat over-engineered approach, especially considering that WZCFG[5] already provides dynamic IP assignment without a DHCP server. Also, no attempt is made to connect users of this system to each other, which makes the experience seem somewhat impersonal. We can consider this application then, as little more than a mobile version of Kazaa (SharmanNetworks, 2004).

2.2.6 Awaire – NYU, USA

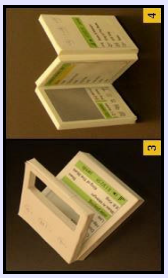


Figure 2-8 *Foam knock-ups of Aura*

The *Aura* is the mobile audio player envisioned by a team at NYU's ITP program as part of their Awaire project. The *Aura* it is said would have high-capacity local storage, use GPS[6] for positioning information, and have broadband always-on internet connectivity. While many of the proposed features are similar to those that have already been implemented in the *tunA* project, the authors also discuss several fresher ideas (Clark, Kantor et al., 2004).

[5] Wireless Zero Config – MS network service which assigns IP addresses without DHCP

[6] GPS – Global Positioning System, a network of US military positioning satellites

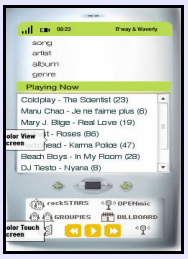


Figure 2-9 *NYU Aura 'screenshot'*

Optimised
Mobile devices have limited memory, small screens/batteries and non real-time operating systems, each of which imposes extra constraints.

One of these is the 'yellow brick road', in which a user sends an invitation to another to find them by following a virtual musical path. If the recipient walks towards the sender they hear one subset of audio, for instance jazz music; similarly walking in the opposite direction triggers the playback or another subset of audio, say heavy metal. It can be thought of as a modern day re-invention of the children's game 'Warmer/Colder', and bears many similarities to the 'Guide Shoes' work at MIT (Nemirovsky & Davenport, 1999).

Intriguing though the ideas may be, the project is entirely conceptual, and the deliverables are 'vapourware'. The team estimates that it will be years before the required infrastructure is sufficiently commonplace for development to commence.

2.2.7 Bubbles – Telenor R&D, Norway

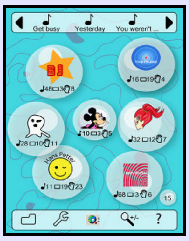


Figure 2-10 *Bubbles' main UI*

Flexible
A graphical interface should be skinnable, buttons should be no smaller than a finger tip, and commonly used controls should be grouped together in a meaningful, intuitive way.

Less ambitious, but with a working prototype is Telenor's *Bubbles* application (Bach, Bygdas et al., 2003). Again running on WiFi-enabled Pocket PC's, *Bubbles* combines the FMOD MP3 engine (Firelight-Technologies, 2004) with the *OpenTrek* ad-hoc networking layer (Sanneblad & Holmquist, 2004). In addition to functioning as a standalone walkman, the application presents users with a list of nearby peers whose audio files are shared using a rudimentary web server.

A key feature of this work is its rich *GapiDraw* based visual interface (Sanneblad, 2004), which represents other users as 'bubbles' consisting of a small graphical avatar and a username. This metaphor is an interesting answer to the challenging design question of how to visualise the complex

Figure 2-11 *Bubbles – showing a download in progress*

and dynamic environment of the user as others fall in and out of range, without comprising the functionality of the software, as a standalone player.

While the GUI[7] may be compelling, the underlying implementation is less so. When a user selects a peer they are presented with a list of the audio available on that individual's device, from which they can select one or many tracks to download over HTTP[8]. There are two distinct disadvantages to this approach. First, with this form of unicast[9] communication each client demands its own stream of data, linearly increasing the demand on both bandwidth and processing power with each additional user. Secondly, by allowing users to download audio other than that which the host is themselves listening to, any feeling of shared experience or bonding is lost, and the encounter is reduced to an anonymous file swapping session.

2.2.8 Sound Pryer - Interactive Institute, Sweden

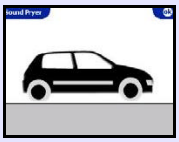


Figure 2-12 *Sound Pryer with peer-car in range*

Sound Pryer is *tunA*'s closest relative. A Pocket PC application written by the Mobility group at Sweden's Interactive Institute, *Sound Pryer* is a mobile audio player that functions as a normal walkman until another peer is discovered, at which point, the two units perform a kind of 'rock-paper-scissors' operation to elect one a master. For the duration of the encounter, the master's audio stream is played on both devices. This is accomplished using custom ports of

[7] GUI – Graphical User Interface

[8] HTTP – Hyper Text Transfer Protocol

[9] Unicast – One-to-one communication / Multicast – one-to-many communication

Figure 2-13 *SoundPryer during playback*

Hi-Fidelity
Core elements of the users experience should never be downgraded. In this instance the streamed media should be of at least comparable quality to that from a local source.

the open source MAD MP3 decoder (Leslie, 2004), and the Live RTP[10] stack for streaming (Ray Finlayson, 1999).

While they bear several similarities, *tunA* expands on *Sound Pryer* in several areas. Firstly, it offers several additional features such as the ability to 'bookmark' interesting audio or peers, an integrated instant-messenger with a custom input mode, and an extensive reporting system to log all system/user activity. Secondly, it offers high-fidelity audio, as opposed to the choppy, low-quality audio reported during user trials in Sweden (Esbjornsson, Juhlin et al., 2003). Thirdly, rather than arbitrarily decide which of two bonded peers should be the dominant audio source, *tunA* makes listening to other peoples music optional, and supports one-to-many audio streaming. Finally, the *Sound Pryer* project is exclusively focused on in-car prototypes, whereas *tunA* is intended to be used anywhere a normal walkman would be.

[10] RTP – Real-time Transport Protocol, A set of end-to-end network transport functions running over UDP, suitable for apps involving real-time streaming of data such as audio/video.

2.3 Social Networks

This section examines why *tunA* was created – as a tool for studying social networks, and encouraging new patterns of group interaction and behaviour. As Brown puts it (Brieger, 2002):

> *"Social structure becomes actually visible in an anthill; the movements and contacts one sees are not random but patterned. We should also be able to see structure in the life of an American community if we had a sufficiently remote vantage point, a point from which persons would appear to be small moving dots. . . . We should see that these dots do not randomly approach one another, that some are usually together, some meet often, some never. . . . If one could get far enough away from it human life would become pure pattern."*

A brief look at online communities is followed by a critique of more similar applications such as *Serendipity* and *HocMan* which also detect the presence of other devices, and use this information to transmit business cards, websites, sense of presence, and viral messages.

2.3.1 Friendster, Orkut, Tribes – Various, USA

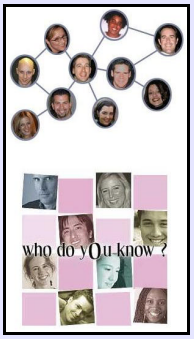


Figure 2-14 *Socialising with Friendster / Orkut*

While social networking websites have existed for years (Barabasi, 2002), a slew of new sites have recently emerged to help people leverage their social networks for help with their careers (*Ryze.com* and *LinkedIn.com*), dating (*Friendster.com*, *Orkut.com*), and recommendations and listings (*Tribe.net*). All of these offer much the same service, allowing people to: explicitly articulate their social network, present themselves through a profile of interests and demographics, post public testimonials about one another, and browse a network of people on the assumption that friends-of-friends are somehow more likely to be suitable dates/employees than complete strangers (Boyd, 2004).

Subvertible
Users sometimes make the best designers, as evidenced by the number of plug-ins, scripts, hacks and modifications available on the Net; the entire open source movement is a testimony to this. Flexible, systems leverage this

An interesting development in the *Friendster* world has been the emergence of 'fakesters', cultural or community characters with whom many people connect such as 'God', 'George W. Bush', 'Manchester United', or 'MIT'. While many users value these fake characters as a means to easily find other people with similar interests, the company has never approved of them, and actively deletes them; for fear they collapse the network, devaluing links that bind users together.

2.3.2 Lovegety / Gaydar – Bandai, Japan

Figure 2-15 *The 'LoveGety', popular in Japan*

One of the earliest works in this field was the *Lovegety*, a small key fob that periodically emitted an RF pulse to any neighbouring device within a thirty foot radius (Jazzbo, 1999). Available in pink or blue, and massively popular in Japan they vibrated or beeped when a *Lovegety* of a different colour was close by. The *Gaydar* was the homosexual equivalent.

Accountable
Although simple low bandwidth data such as presence or status can be powerful, most trends only emerge over a longer period of time. A mobile application should log every internal and external event for later analysis.

That millions of these units were sold is a compelling demonstration of the appeal of any technology which can help introduce us to strangers, (admittedly though, no data on the relative sales of male/female units is available). The *Lovegety* is however a very coarse grained tool, transmitting as it does only three bits of information: gender, presence, and state of arousal. Selecting additional data that is both interesting to others, yet at the same time non-identifying, is a challenging task.

2.3.3 Spot Me – Shockfish, Switzerland

Figure 2-16 *Shockfish's SpotMe device*

Independent *Wherever possible existing, ubiquitous infrastructure such as the mobile phone network should be used instead of a bespoke solution. Dependence on repeaters, base stations, or kiosks severely limits the range, and use of an application.*

The *Spot Me* is a small battery-powered handheld computer running embedded Linux (Shockfish, 2004). It incorporates a small LCD display, a few buttons, and both RF and IR connectivity. It is primarily used at large meetings or conferences to exchange electronic business cards, and advise delegates of schedule updates.

The more compelling uses of these devices however, are their built in 'spot' and 'radar' features which automatically alert users when they are within a few feet of another attendee with whom they have either expressed an interest in meeting, or with whom they share common interests. From the success of these devices, we can infer that people are willing to sacrifice the privacy of personal information (including their location) where they perceive value in the reward for doing so, in this case an increased opportunity to network with peers. More critically however, these devices are expensive to deploy, require an extensive infrastructure of RF repeaters to support them, and may be seen by some as being slightly fiddly or cumbersome to use, especially for text entry.

2.3.4 Hocman – Interactive Institute, Sweden

Figure 2-17 *A Hocman website*

Hocman is a small Pocket PC application for motorcyclists. As they pass each other on the highway the devices swap simple web pages containing a short description of their owner's bikes. The application itself is fairly straightforward, a mobile HTTP 1.0 web server with a simple peer discovery mechanism, and statically assigned IP addresses. Although little more than contact information is exchanged during an

encounter, this is more a function of the tight time constraints involved when two bikers meet each other on a motorway than anything else.

Brief
Encounters are often short, and can be terminated by either party at any time. Proximity apps must handle these conditions gracefully.

Feedback from users in several trials (Esbjornsson, Juhlin et al., 2002) suggests that there is an interest from the motorcycle community in similar bond-enabling applications. In addition to this, the researchers noted that it was the simplest features, such as the audible chime the software emitted when another bike was detected, that the users seemed to find most appealing.

2.3.5 MemeTag, UberBadge - MIT Media Lab, USA

Figure 2-18 *The MemeTag, an early 'smart' badge*

Figure 2-19 *The UberBdage from the MIT MediaLab*

For years now, a recurring theme at the MIT Media Lab has been the iterative development of 'smart' badges to relay and swap information with others. In the past they have been used to track the spread of information, observe interactions between conference attendees, and relay context-sensitive information to either the wearer, or the person facing them.

Early work revolved around the *Thinking Tag* (Borovoy, Martin, Resnick et al., 1998) and the *Meme Tag* (Borovoy, Martin, Vemuri et al., 1998). These were small battery powered wearables with bidirectional IR, a backlit LCD display, and a few buttons for input. Conference participants used them to exchange 'memes' (succinct ideas or opinions) with each other. These snippets were uploaded to the badges at terminals scattered throughout the building, and were beamed from tag to tag via infrared. Large screens tracked the progress of the 'memes' from person to person, and displayed visualisations of the social activity at the event.

Featuring three processors, RF and IR communication, a built-in microphone, speaker, large LED and LCD displays, a pager vibrator, a flash memory connector, and several expansion slots, the *UberBadge* is the latest incarnation of this line of work (Laibowitz & Paradiso, 2003). Again the badge is designed to be worn around the neck at conferences, and used for viral messaging, locating attendees, and profiling social interaction.

Novel
There is always a temptation to develop an in-house hardware platform, but in most cases there exists a cheap off the shelf unit that offers the same functionality.

While the idea of studying behaviour patterns is an interesting one, the value of the studies conducted to date is unclear. The limited functionality of the *Meme Tags* meant that findings were coarse-grained and few non-obvious trends emerged (e.g. exhibitors speak to more people than attendees). While the *UberBadge* goes someway to address this, there are no published studies of its use as of yet, and from an engineering perspective it seems to be a re-invention of the wheel – a standard iPaq would be smaller, lighter, and more powerful.

2.3.6 Serendipity – MIT Media Lab, USA

Figure 2-20 *Serendipity*

Nathan Eagle at the MIT MediaLab has been working on *Serendipity*, a lightweight application for Symbian[11] mobile phones that uses Bluetooth[12] to emit a unique ID over a 16' radius (Eagle, 2004). If two phones come within range of each other, their IDs are sent to a central server, which looks up their profiles. If a match is found, each gets the other's name, thumbnail photo, and a short list of common interests sent to their device.

[11] Symbian – Joint venture by Psion, Motorola, Nokia and Ericsson to develop the EPOC O/S

[12] Bluetooth – Short-range (10m) wireless (2.4GHz ISM) comms protocol for mobile devices

By leveraging the ubiquitous mobile phone network for communication, *Serendipity* is freed of the infrastructure requirements that constrain the *Spot Me* device. The client software is also very straightforward, making it easily portable to alternative operating systems such as WinCE.

Private
Any application which transmits data to a central server will aggregate a large amount of information, which may pose a threat to the privacy of its users. This threat increases exponentially as information from multiple sources is cross-correlated.

While offering more information than the *Lovegety* and a greater range than the *Spot Me* there are areas where it could improved upon. For instance, while it does a solid job of peer discovery, the use of central server raises a number of privacy concerns, as the logs of this server may reveal detailed information about the movements of users over time. A combined praise / criticism of this system, is that this data could potentially be mined to infer the nature of relationships between users on the basis of the frequency and length of their interactions.

2.4 Shared Experience

By allowing strangers to discover each other and 'jack in' to one another's audio streams, *tunA* creates a shared audio experience between people; this is how it fosters social connections. This section reviews other technologies that use shared activity as a means of bonding, starting with the relatively banal examples of file and play list sharing, and moving on to more related projects such as *Breakout for Two*, *Reflexion*, and *iCom*, all of which make use of this ploy. The discussion concludes with Sony's *Music Maker*, an application for distributed, collaborative musical creation.

2.4.1 iTunes – Apple Computer, USA

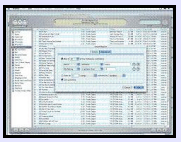


Figure 2-21 *Apple's iTunes music library with play list sharing*

Interesting *People's curiosity adds 'stickiness' to sites, devices and applications. The most seemingly banal information (blogs, status lines, play lists etc.) can arouse great interest.*

iTunes is a highly popular online software store and associated player. Among its more interesting features are the ability to share music with up to five other users, and the power to stream audio to speakers in another room using an *Airport Express* wireless router (Apple, 2004a).

In addition it can easily extended in *AppleScript*[13] as demonstrated by *StatusPlay*, a plug-in that displays the currently selected track in a users *iChat*[14] status line (Brucker-Cohen & Agamanolis, 2004). Another of example of this is *Tunes at Work* which makes an *iTunes* library available from a remote location over HTTP (TunesAtWork, 2004). However, as commercial software, any of the AAC[15] encoded audio purchased from the *iTunes* store is beholden to a slew of restrictive DRM[16] controls, which severely limit any

[13] AppleScript – Simple, yet powerful scripting language for Mac OS computers.

[14] iChat – Instant messaging client for the Mac OS

[15] AAC – Advanced Audio Coding, an audio encoding scheme employed by Apple.

[16] DRM – Digital Rights Management, software which controls the use of digital media

Figure 2-22 *StatusPlay and iChat*

copying, lending or redistribution of legitimately purchased tracks (Lessig, 2001). Furthermore there is no 'eavesdrop' feature meaning that while it is possible to share audio among users, there is no facility to listen to audio together. This particular criticism it must be said, is shared with the majority of this type of software, and is one the more innovative features of *tunA*.

2.4.2 Napster / Kazaa / Aimster - Various, USA

Figure 2-23 *The Napster Logo, now a cult icon*

Napster changed the music industry forever. The idea was simple - allow users to share and swap music online. The flaw which led to its eventual demise however, was that although files were transferred in a 'peer-to-peer' fashion, that is directly from one machine to another, search requests were filtered through a central server which was eventually crippled and shutdown by the US courts (Oram, 2001).

It was not long before several other services like *Gnutella*, *Kazaa*, *Morpheus*, *LimeWire*, and *BearShare* took over the file-sharing mantle, and offered improvements such as truly distributed searches, split-downloads from multiple sources, integrated virus scanners and the ability to share more than just music (Menn, 2003). Among these, one in particular stands out for its slightly unusual approach. *Aimster*, an extension of AOL's instant messenger, also allows users to share their music libraries, but only to those on their 'buddy list'. Renamed *Madster* after a legal squabble with AOL, it forms something of a bridge between anonymous file sharing, and the more personal world of instant messaging.

Figure 2-24 *Screenshot of the Aimster plug-in for AOL's AIM*

2.4.3 ShoutCast / IceCast – Nullsoft, USA

Figure 2-25 *ShoutCast GUI*

Bidirectional *A one-way exchange offers an incentive to participate only to the consumer. Providing a mechanism for feedback can help to redress this imbalance.*

ShoutCast is a plug-in for the *WinAmp* media player that in effect turns it into an Internet radio station, supporting unicast streaming audio over HTTP (Papdakis & Douligeris, 2001). It supports multiple formats, and audio can be live, pre-recorded, or even from a line-in source such as a microphone. By running over HTTP it can reach the majority of hosts on the Internet, and an open source alternative *Icecast* is readily available for download.

While several of these features appear attractive at first glance, what compelling reason is there to avail of them? How does one choose between the thousands of streams registered in the *ShoutCast* directory? The experience is not a collaborative one - there is no opportunity for listeners to give feedback or to communicate with each other, so why bother?

2.4.4 Breakout for Two - Media Lab Europe, Ireland

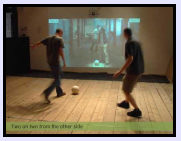


Figure 2-26 *Breakout for Two*

Shared *Mutual activity, be it playing football or just getting drunk together, forms a common frame of reference which can encourage discourse and bonding.*

Breakout for Two is a multiplayer videogame that sports a prototype 'exertion interface', so called because they deliberately require intense physical effort to use (Mueller, Agamanolis et al., 2003). A cross between football, and the arcade classic *Breakout*, contestants at remote locations must break the blocks on a shared video wall by throwing, hitting, or kicking a ball at them. The central tenet of Mueller's thesis is that by adding exertion to a shared experience, the propensity for participants to establish a bond is increased. A control study of a similar system, less the physical interface, would seem to lend substantial credence to the theory. While the addition of exertion was found to increase the trust and

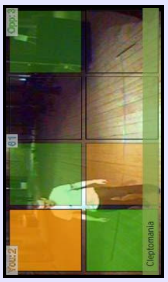


Figure 2-27 *Shared video wall in BreakOut*

goodwill between competitors, both versions demonstrated some success in this effort, suggesting that shared activity itself is a reasonable means of establishing a bond, or 'breaking the ice'.

One possible improvement to *Breakout for Two* would be to scale it to a more open, distributed architecture so that multiple players at distinct sites could organize leagues, much as online gamers do today.

2.4.5 iCom, Reflexion of Presence – Media Lab Europe, Ireland

Figure 2-28 *The iCom, an ambient display at Media Lab Europe*

Ambient *Proximity applications require a large critical mass of users to enjoy any success. Running a service in an unattended or ambient state can go someway to mitigating this.*

Figure 2-29 *Reflexion of Presence*

The *iCom* is a shared media space that connects several points at MITML and MLE, twenty-four hours a day (Agamanolis, 2001). Each installation consists of a large video-projection screen, at least one camera, and a set of echo-cancelling speakers/microphones. The displays at each site are synchronised, and show a user-configurable mixture of video feeds, and any emails sent to an internal mailing list.

iCom can be used as a conventional video conferencing system for ad-hoc meetings and collaborations, but its more powerful application is as an ambient display. The shared video link between remote locations provides a continual background awareness of the daily habits and routines of distant peers. *Reflexion* is a video communication system that operates like a 'magic mirror' in which you see reflections of both yourself and other participants at remote locations, overlaid on a static background or live video feed (Cullinan & Agamanolis, 2003). By monitoring visual and auditory cues, the system detects the dominant speaker and emphasises them by partially fading out the other users.

2.4.6 Music Maker – Sony CS Lab, France

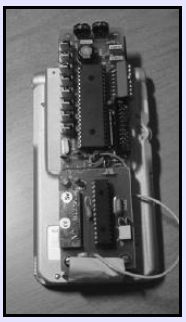


Figure 2-30 *Sony's tool for collaborative music making*

Music Maker is a system for distributed, collaborative musical creation (Tanaka, 2004). It consists of three main components: a Sony Clie[17] handheld augmented with force sensing resistors (FSR) and accelerometers, a generative music engine, and a suite of network services that manage authentication and media delivery. With these tools, multiple users can create and listen to audio together.

Tanaka uses an interesting public-key encryption system for user management, based on friendship and introduction (Perkins, 2001). When two users become friends, they exchange certificates each signed by the other's public key. When one user goes to a new unknown user introducing himself as 'a friend of a friend', the fact that the two users each have the public key of their common friend enables them to securely verify each other's identities.

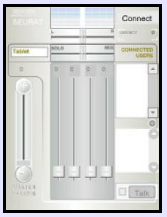


Figure 2-31 *Screen Layout of MusicMaker*

Unfortunately, current prototypes rely on a server running MaxMSP[18] for the audio generation engine, and are equipped with a large ungainly external backpack for the array of FSR's. Both of these shortcomings limit the portability of the system, and constrain it to a lab environment for now. One would expect these prototypes to evolve more fully over time.

[17] Clie – Brand name for Sony's line of Personal Digital Assistants

[18] MaxMSP – Graphical composition package @ www.cycling74.com/products/maxmsp.html

2.5 Design Goals

Many design goals can be derived from the features and failings of these related works, some of which are summarised in the left hand column of the preceding sections, and the most important of which are listed here:

- **Synchronous** – The human ear will assume two audio signals are 'coherent' (i.e. from the same source) if they arrive within 30ms of each other. This is the approximate level of accuracy required to allow multiple users to 'bop along' to the same piece of music (e.g. *MyFi*).
- **Engaging** – Personal technologies should not isolate their users from their surrounding environments as most existing devices do, but instead should encourage or at least allow engagement with the environment and populace surrounding them (e.g. *Sonic City*).
- **Personal** – Anonymous file swapping, while it may in some cases be a desirable feature, is a poor substitute for personal contact; the fan is more important than the music (e.g. *Pocketster, Aimster*).
- **Optimised** – Mobile devices have limited memory, small screens, short lived batteries, and non real-time operating systems, each of which impose additional constraints (e.g. *Awaire*).
- **Flexible** – A graphical interface should be skinnable, buttons should be no smaller than a finger tip, and commonly used controls should be grouped together in a meaningful, intuitive way (e.g. *Bubbles*).
- **Hi-Fidelity** – Core elements of the users experience should never be downgraded. In this instance the streamed media should be of at least comparable quality to that from a local source (e.g. *Sound Pryer*).
- **Subvertible** – Users sometimes make the best designers, as evidenced by the number of plug-ins, scripts, hacks and modifications available on the Net; a fact to which the entire open source movement pays testimony. Flexible, adaptable systems leverage this to their advantage (e.g. *StatusPlay*, and *Friendster.com*).

- **Accountable** – Although simple low bandwidth data such as presence or status can be powerful, most trends only emerge over a longer period of time. A mobile application should log every internal and external event for later analysis (e.g. *Lovegety*).
- **Independent** – Wherever possible, existing, ubiquitous infrastructure such as the mobile phone network should be used instead of a bespoke solution. Dependence on repeaters, base stations, or kiosks severely limits the range, and use of an application (e.g. *Spot Me*).
- **Brief** – Chance encounters are often short, and can be terminated by either party at any time. Proximity apps must handle these conditions gracefully, for example by using streams instead of trying to exchange large, atomic chunks of data before engaging each other (e.g. *HocMan*).
- **Novel** – There is always a temptation to develop an in-house hardware platform, but in most cases there exists a cheap off the shelf unit that offers the same functionality (e.g. *UberBadge*).
- **Private** – Any application which transmits data to a central server will aggregate a large amount of information, which may pose a threat to the privacy of its users. This threat increases exponentially as information from multiple sources is cross-correlated (e.g. *Serendipity*).
- **Interesting** – People's curiosity adds 'stickiness' to sites, devices and applications. The most seemingly banal information (blogs, status lines, play lists etc.) can arouse great interest (e.g. *iTunes*).
- **Bidirectional** – A one-way exchange offers an incentive to participate only to the consumer. Providing a mechanism for feedback can go some way to redressing this imbalance (e.g. *ShoutCast*).
- **Shared** – Mutual activity, be it playing football or just getting drunk together, forms a common frame of reference which can encourage discourse and bonding (e.g. *Breakout for Two, Relfexion*).
- **Ambient** – Proximity applications require a large critical mass of users to enjoy any success. Running a service in an unattended or ambient state can go someway to mitigating this (e.g. *iCom*).

2.6 Summary Remarks

The second chapter brought the discussion forward by presenting a literary review of related research in three main areas: Mobile Audio, Social Networks, and Shared Experience, or in other words, the what, why, and how of this thesis. Under each of these headings, several projects were examined, beginning with those bearing the least direct relationship to the idea of a socialising walkman, and ending with those exhibiting the most likeness. Throughout this process a series of design goals were derived, and these were synopsised at the end of this section. In the final chapter these will be revisited in light of the end deliverable: a functioning prototype software package.

In the meantime, the next two chapters will discuss the specific implementation of this application, and detail several preliminary studies evaluating its use in the field.

3 Design / Implementation

3.1 Demo or Die

As the MediaLab community is heavily dependent on private industry to sponsor its research programme, much of the technology transfer that takes place occurs in the form of regular technology demonstrations to partners (Haase, 2000). This in turn has led to the evolution of a 'demo or die' culture which substantially influences the way in which projects evolve, driving them into a iterative design cycle with short epochs, that invariably begins with a 'hacked' proof-of-concept design. *tunA* was no exception to this prime directive.

Initial versions operated on a fixed-infrastructure platform, with the experience of shared audio being simulated by a central server which co-ordinated timing using standard client-server techniques (Tanenbaum, 2003) and identical volumes of audio on each device. Later experiments saw the code-base transition to a peer-to-peer model for device discovery, but with a reliance on local sources of audio, exchanging 'play track x at pos y' style commands to synchronise streams. Eventually the project matured into the truly peer-to-peer, scalable, 'hack-free' version whose innards are the focus of the remainder of this chapter. These next sections detail the precise requirements, specifications, architecture, and implementation of the *tunA* system and the set of custom libraries on which it draws.

3.2 Functional Overview

3.2.1 Minimum Requirements

Current software installations run on the 802.11b enabled HP iPaq 415x range (see Appendix H), chosen for their combination of low unit cost (less than $500), integrated WiFi adaptor, and slim form factor. It is however designed to run on any WiFi equipped Pocket PC device running Windows CE.Net 4.2 or above, and could be readily extended to function over another wireless standard, such as BlueTooth, or with some modifications, another operating system such as Linux. The application and its associated libraries and graphics require less than 1MB of storage space on a release build, and at least 8MB of free program memory to run effectively. All content is stored on removable flash memory (MMC/SD cards), the size of which is at the users discretion.

3.2.2 Specifications / Features

Influenced by the design goals derived from the earlier literary review, and preliminary user studies of chapter four, *tunA* sports the following features, each of which is more fully discussed in the next section:

- **MP3 Decoding** – Local and streamed playback of any CBR encoded MPEG 1.0 Layer 3 audio file, provided by a customised version of the integer-based FMOD audio library.
- **ID3 Parsing** – Full support of the ID3v11 specification for storing metadata such as the artist, title, genre, track number etc. directly in the audio file. Partial support of the ID3v2 standard.
- **Instant Messenger** – Simple chat protocol for sending text, binary files, and profile data to nearby peers.
- **Synchronisation** – Heuristics to synchronise streamed audio between multiple distributed devices for a 'shared audio experience'.
- **T9 Engine** – Port to Windows CE of fellow MediaLab researchers UNIX-based predictive text algorithms, for faster user input.

- **Discovery Service** – Algorithms and code to rapidly detect nearby peers and establish an Ad-Hoc network.
- **Skinnable UI** – Set of custom graphical widgets based on owner-drawn MFC controls for building rich user-interfaces on WinCE.
- **SQL D/B** – Logging mechanism which records details of all system activity, both to a SQL database, and to a flat file system.
- **Networking** – Bespoke networking library which provides reliable UDP multicasting, TCP/IP server support, SSID manipulation, and asynchronous socket support, all of which are lacking as standard.
- **Utilities** – Various utility classes offering support for Unicode files, registry entries, system notifications, accurate timing data, control of H/W keys, retrieval of system serial numbers, generation of GUID's, dynamic linking of libraries and so on

The primary means of communication between the different subsystems is windows message handling. Where there are critical timing requirements UI threads with individual message pumps are used, otherwise the messages are trapped and dispatched by the pump in the main dialog.

3.3 System Architecture

The overall architecture of the *tunA* platform is illustrated in two diagrams. The first, Figure 3-1 shows how individual devices communicate with other peers on the same Ad-Hoc network in two ways: a UDP multicast channel shared between all of them, and separate each-to-each TCP/IP connections between all peers.

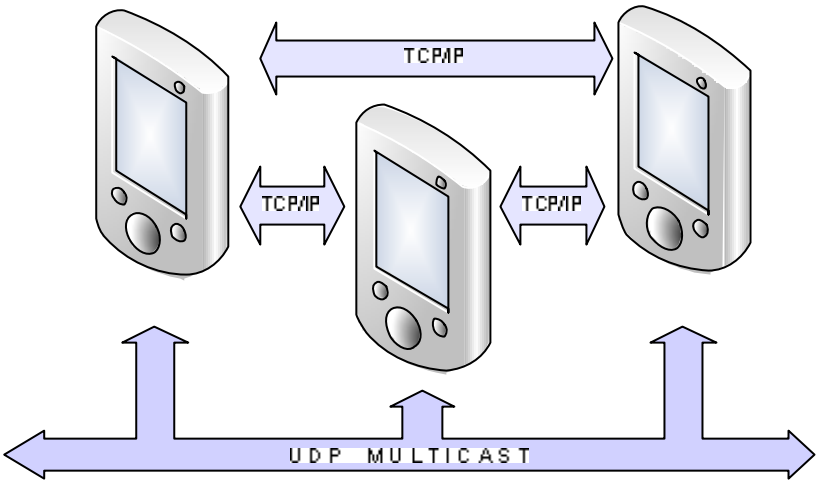


Figure 3-1 Interaction of multiple peers

The second, Figure 3-2 shows an expanded view of a single *tunA* peer, and the interaction between each of the major subsystems. Each of these systems is labled with the section number that contains a more indepth view of them.

Broadly speaking, *tunA* peers discover each other by periodically multicasting UDP packets announcing their presence to all nearby devices, and maintaining a list of those peers from whom it has detected similar packets within a specified time. When a user selects a local audio track, the system begins to multicast packets consisting of some timing info, and frames of MP3 data to all interested peers (itself included.) A separate 'listening' process marshals this data into a buffer from which the MP3 decoder reads. The timing info is used to regulate the contents of the buffer, and the requests of the decoder to provide a synchronised audio experience between peers. The IM component

exchanges profile data, text messages, and graphical avatars over separate TCP/IP connections. A database maintains a record of all peers, events, audio, and messages encountered by the system.

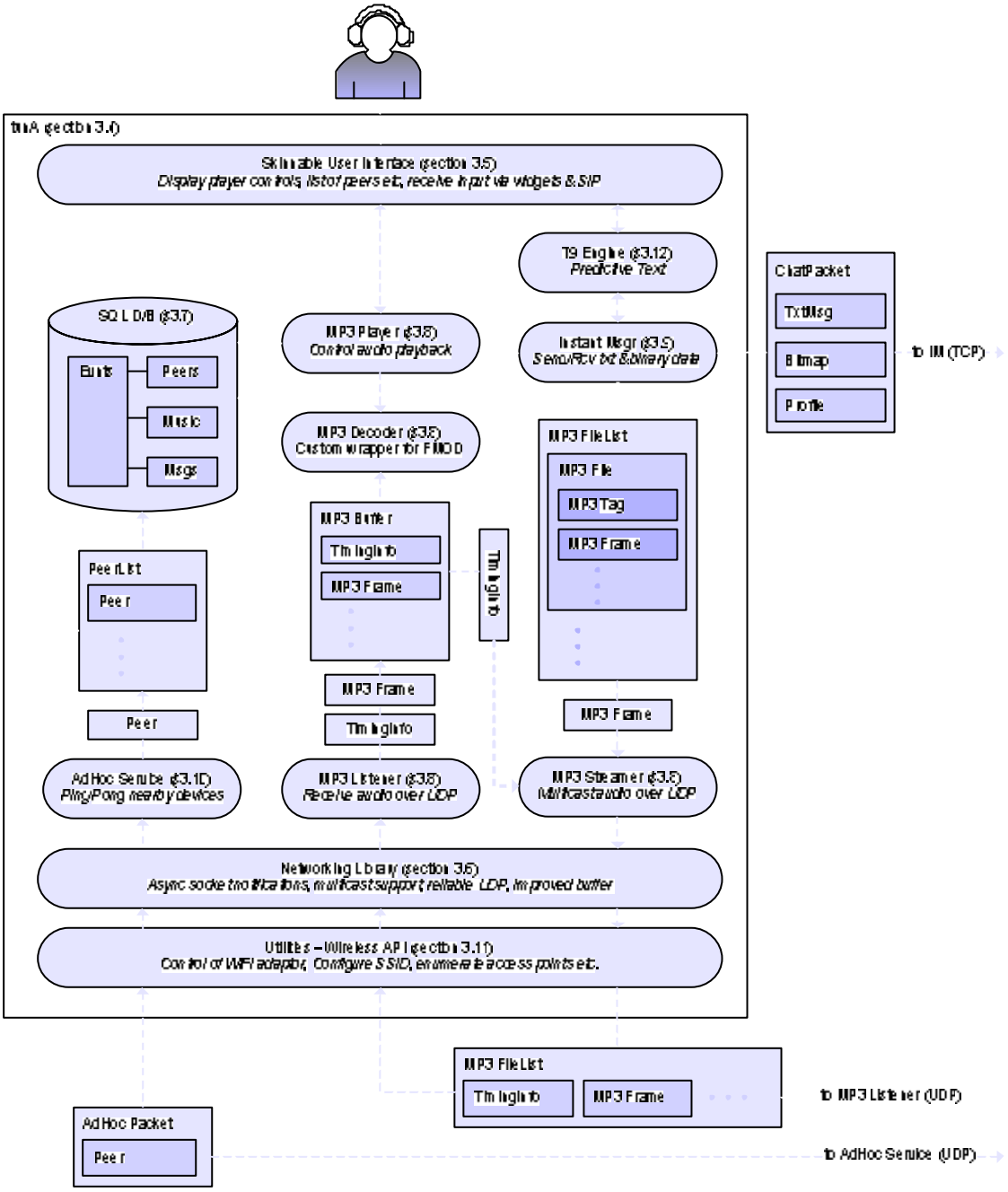


Figure 3-2 Block architecture diagram of a single tunA peer

The remainder of this chaper, consists mainly of a more indepth look at each of the subsystems refered to in the above diagram, begining with the main app.

3.4 Application

tunA is an MFC[19] dialog-based application for the Pocket PC platform, which itself is an image of the Windows CE.Net operating system. This subsection deals with the main message handling loop, and its associated classes.

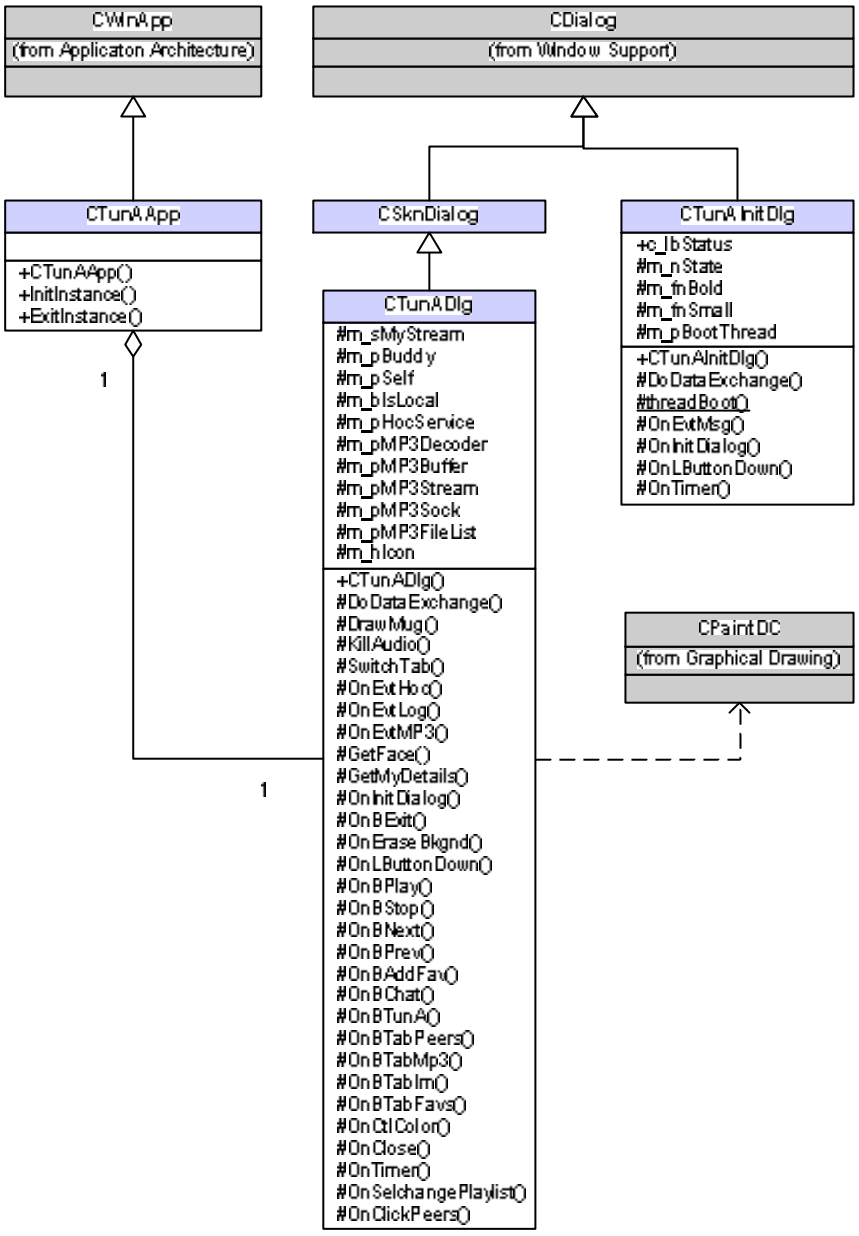


Figure 3-3 UML Application Class Diagram

[19] MFC – Microsoft Foundation Classes, a library of C++ classes which provide commonly used containers (e.g. arrays, lists etc.) and wrappers to many underlying WinCE structures.

Figure 3-3 above is a UML[20] class diagram which formally describes the main application classes, and their relationship to the underlying code from which they are derived. Grey boxes represent generic MFC classes, whereas those with a blue title are *tunA* specific. A solid line with an arrow indicates *inheritance*, or the presence of an 'is-a' relationship (e.g. a circle 'is-a' type of shape); a solid line with a white diamond denotes *aggregation*, or a 'has a' relationship (e.g. a car 'has a' steering wheel); a solid line with a black diamond shows *composition*, a stronger form of the 'has a' relationship where the lifetime of the composed object is directly dependent on the other (e.g. a body 'composes' a brain, when the body dies so does the brain); and a dashed line shows *dependence*, or a 'depends on' relationship. If a class is not fully expanded, (such as *CSknDialog* above), it signifies that it is explained in another more appropriate section of the text. Finally, these diagrams are only intended to give an indication of how classes in a given subsection interact with each other. As such, arguments, types, and initial values, have all been suppressed (see Appendix K for more specific details).

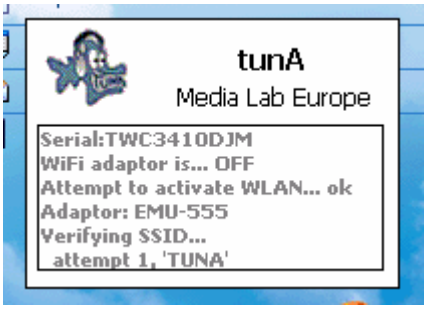
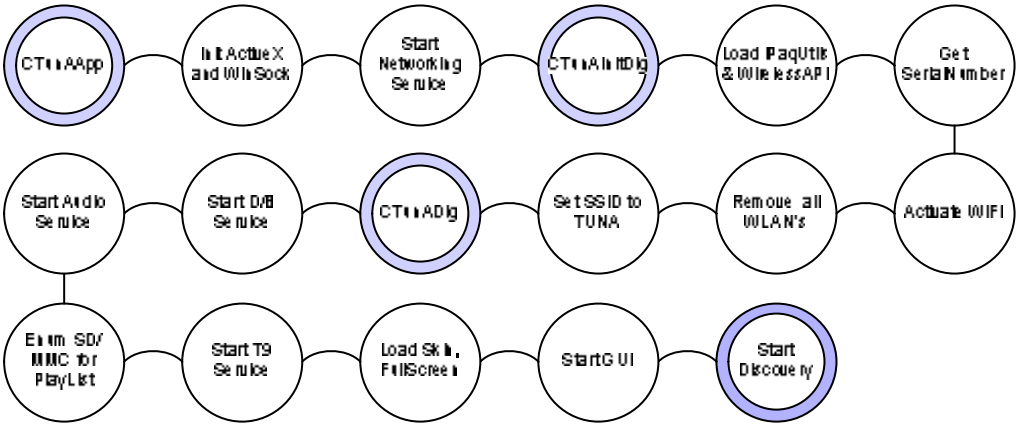


Figure 3-4 CTunAInitDlg, the initial splash screen

CTunAApp is the base class of the application, the first created, and the last to be destroyed. As well as performing some basic initialisation and cleanup, it creates one instantiation of the *CTunADlg* class, which contains the main

[20] UML – Unified Modelling Language, which programmers use to formally and diagrammatically codify methodologies and processes in an easy to understand fashion.

message handling loop, and all of the graphical controls (ListBoxes, Buttons and so on) that the user interacts with. This dialog handles all the user interaction events such as clicking a button, scrolling a list, or pressing a key, and passes messages on to the other subsystems on the basis of them.



tunA is a multithreaded application, which has the advantages of making much of the code easily reusable and much more efficient, at the expense of complexity. To provide a clearer picture of how these systems interact with each other, Figure 3-6 below presents the UML Sequence diagram of a typical use case - the user pressing PLAY.

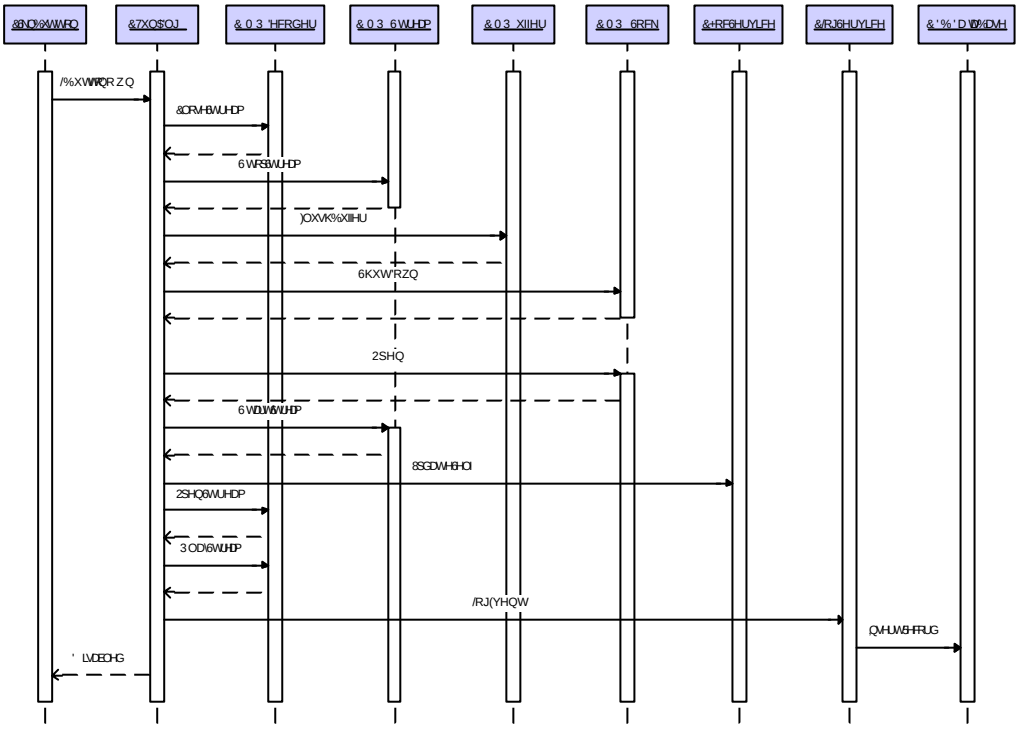


Figure 3-6 Sequence diagram showing a Play event

The horizontal dimension represents different actors or objects, with their life spans indicated by vertical blocks; the vertical dimension represents time, with time proceeding down the page. Arrows represent messages between objects, with return values as dashed lines. In this instance, when the play button is pressed, it sends a message to main dialog, which kills audio playback, stops the streamer from reading blocks from storage, flushes the buffer, kills the old listening socket and creates a new one, starts the streamer again, waits for the buffer to fill, then starts audio playback. At this point the playback event is sent to the logging service, which asynchronously creates a record of this in the database, and the play button is disabled.

3.5 User Interface

tunA sports a full-screen, skinnable user interface, implemented as a set of sub-classed owner-drawn MFC controls (ListBoxes, RichInks, Buttons, Edits, Statics etc). By supplying a set of BMP/GIF images, and an ASCII text file describing their location, content and attributes, a user can modify the appearance of these graphical widgets.

Figure 3-7 Screenshots of tunA's default skin

It is not obvious how best to configure the interface in a portable peer-to-peer application such as this, primarily because until market penetration reaches a certain critical mass, the device will often have to function as a standalone player. The requirements for these two modes of operation are somewhat different. For instance, the instant messenger, list of nearby peers, and favourites features are of little use when there are no other peers nearby. The layout is also constrained by the size of the user's fingertips.

The skin depicted in Figure 3-7 above is the result of several user studies discussed in the next chapter. By default the UI is split into four tabbed screens, divided by functionality. The first displays a list of all other tunA peers within range; the second controls local music playback; the third features an instant messenger with an input method similar to a mobile phones SMS system arranged in parallel on the vertical sides of the screen; and the fourth contains a series of options, and a list of all songs/people marked as 'favourites'. These widgets are all familiar windows controls, whose draw/paint methods have been over-ridden to provide a richer graphical appearance. One such is example is given in Extract 3.1 below:

Extract 3-1 – CSknButton::DrawItem(LPDRAWITEMSTRUCT)

```
// Determine button state
BOOL bIsPressed  = (lpDrawItemStruct->itemState & ODS_SELECTED);
BOOL bIsDisabled = (lpDrawItemStruct->itemState & ODS_DISABLED);
BOOL bIsChecked  = (lpDrawItemStruct->itemState & ODS_CHECKED);
BOOL bIsChecked  = GetChk();

// Select appropriate image (down, up, focused, disabled etc.)
CBitmap* pNewBmp = NULL;

if( bIsDisabled )                // DISABLED
    pNewBmp = &m_bmpDisabled;
else
{
    if( bIsChecked )             // CHECKED
        pNewBmp = &m_bmpDown;
    else
    {
        if ( bIsPressed )        // DOWN
            pNewBmp = &m_bmpDown;
        else
```

```
            pNewBmp = &m_bmpUp;    // UP
        }
    }

    // Create compatible device context
    CDC *pDC = CDC::FromHandle( lpDrawItemStruct->hDC );
    CDC memdc;
    VERIFY( memdc.CreateCompatibleDC( pDC ) );

    // Make sure we've loaded this bitmap
    ASSERT ( pNewBmp != NULL );
    CBitmap *pOldBmp = memdc.SelectObject( pNewBmp );

    // Get bitmap dimensions
    BITMAP bmInfo;
    pNewBmp->GetBitmap( &bmInfo );
    CRect rect;
    GetClientRect( &rect );

    // Strech image to cover control
    pDC->StretchBlt( 0, 0, rect.Width(), rect.Height(), &memdc,
        0, 0, bmInfo.bmWidth, bmInfo.bmHeight, SRCCOPY );

    // Restore old bitmap
    memdc.SelectObject( pOldBmp );
```

Figure 3-10 (on the last page of this section) demonstrates the class hierarchy in the graphics subsystem. *CSknDialog*, the main class is derived from a standard windows dialog and it is to this that all the other controls are bound. It adds support for full screen behaviour (using the SHFullScreen API), replaces the default white background with a bitmap (by handling the WM_ERASEBKGND event), and disables the 'Red Dots' animation that WinCE uses to simulate a right-click (see Figure 3-8 below).

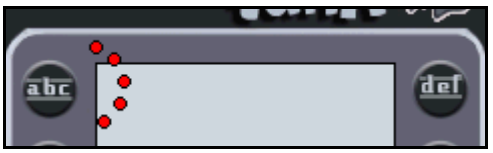
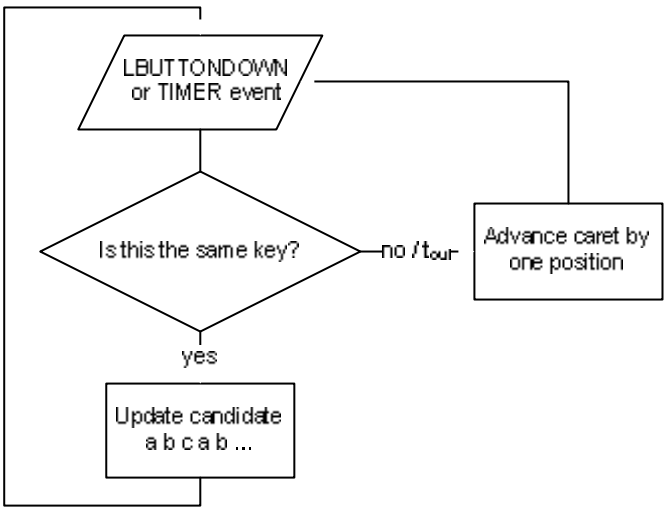


Figure 3-8 'Red Dots' Animation

CMobEdit is an edit control that has been modified to behave like the text input screen of a mobile phone. It provides a flashing caret that delays on each

character as it is entered, as codified in Figure 3-9 below. This behaviour changes slightly when the T9 engine is engaged (see Section 3.12).



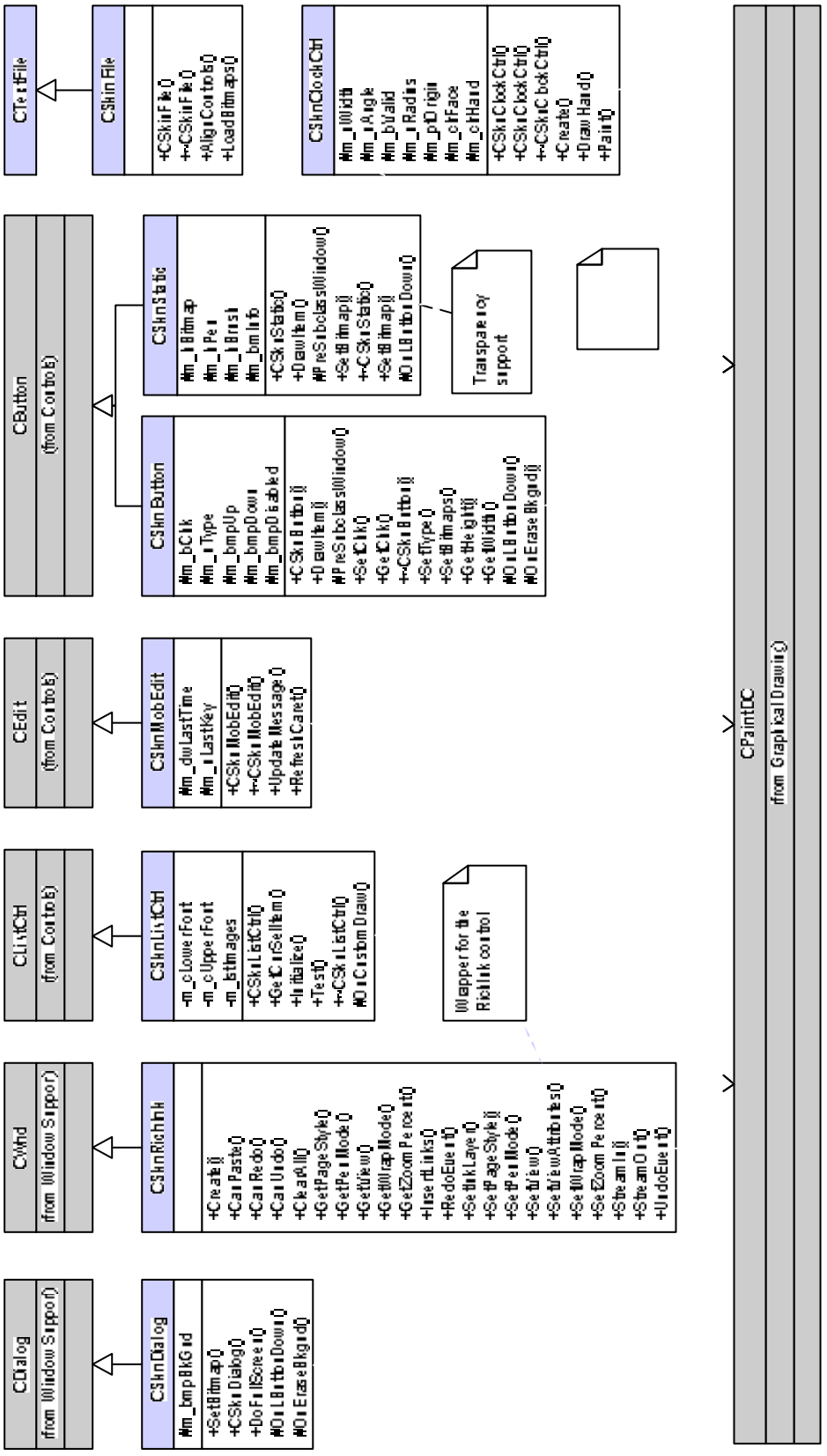
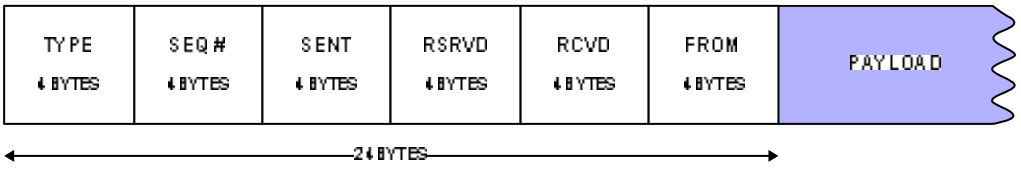


Figure 3-10 UML GUI Class Diagram

3.6 Networking

Networking support on the Pocket PC platform is limited, and riddled with bugs, undocumented features, and inconsistencies. The UDP buffer has a default size of two, so that if multiple packets arrive in a short space of time applications suffer from massive packet loss (incidentally the only way to correct this is to first *add* and then modify an undocumented registry key). Asynchronous sockets are not supported, and the 'hacked' *CCeSocket* class in the MFC library that attempts to emulate this feature not only breaks the MFC inheritance hierarchy, but introduces several bugs of its own, including lack of listening stream socket support, and several 'ghost' notifications which are never sent (Makofsky, 2004). Furthermore, performance varies drastically from device to device, even when they have equivalent specifications. Hewlett Packard's top of the range 555x series for instance, despite boasting superior overall performance is greatly outshone by the 415x range. Overall, the support provided from the MFC library in particular is so weak that it was necessary to write a library of C++ classes to re-wrap WinSock from scratch.

At this point it is useful to describe the various classes that make up this replacement networking library, all of which are presented in Figure 3-12 overleaf. *CNetSockAddr* wraps the SOCKADDR_IN structure and consists mainly of helper functions and operators to make it simpler to access and mutate socket addresses (IP addresses, port number etc.) *CNetPacket* is a special type-safe array of bytes that can dynamically shrink and grow as required, each of which has an associated header contained in *CNetHeader*, the structure of which is shown in Figure 3-11 below:



The first field stores the type of packet (MP3 data, IM, peer discovery), the next the sequence number, followed by t_{Sent} (the logical time at which it left the sender), a reserved field for extra timing information, t_{Rcvd} (the logical time at which the packet is received), and lastly the IP address of the sender.

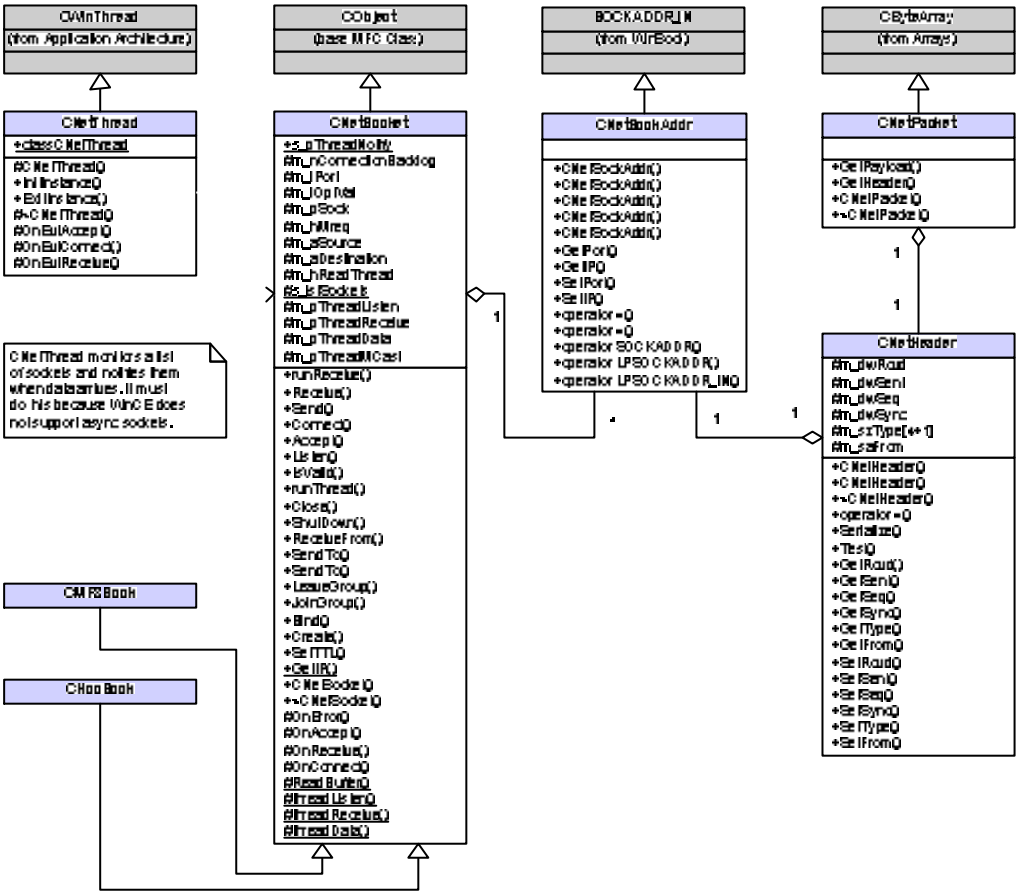


Figure 3-12 UML Networking Class Diagram

The next two classes require more explanation. *CNetSocket* is the main workhouse, providing support for both stream and datagram[23] sockets. However the class also contains a static list of pointers to every instance of the class, and a static member function which runs as a separate thread, selecting on this list of sockets until data arrives at one of them. When this occurs a

[23] In stream communications, a dedicated connection is setup between the parties involved (e.g. phone call, TCP/IP). Datagram sockets on the other hand are stateless, requiring no setup, but message delivery is not guaranteed, only aimed for (e.g. UDP).

message is sent to another thread, *CNetThread*, which in turn calls one of the callback functions *OnAccept()*, *OnConnect()*, or *OnReceive()* depending on the particular event. These functions are all declared as virtual, so in practice to create an asynchronous socket, one simply subclasses *CNetSocket* and overrides these functions to receive callbacks, as demonstrated in Extract 3-2 below.

Extract 3-2 – Overriding socket function for notifications

```
void CMP3Sock::OnReceive( int nErrorCode, CNetPacket *pPacket )
{
  #ifdef MP3SCK_DEBUG
    TRACE( _T("CMP3Sock::OnReceive <%d><%d>\n"), nErrorCode, pPacket );
  #endif //MP3SCK_DEBUG

  // Make SURE the socket has been initialised
  ASSERT( m_pBuffer != NULL );

  BYTE *pData = pPacket->GetData();
  int   nSize = pPacket->GetSize();

  // Get Sync number of the PACKET
  DWORD dwSync = MAKELONG (
    MAKEWORD(pData[nSize-20], pData[nSize-19]),
    MAKEWORD(pData[nSize-18], pData[nSize-17]),
    );

  // Get Sequence number of the LAST frame
  DWORD dwSequence = MAKELONG (
    MAKEWORD(pData[nSize-16], pData[nSize-15]),
    MAKEWORD(pData[nSize-14], pData[nSize-13]),
    );

  // Add the frame data to the buffer
  ( (CMP3Buffer *)m_pBuffer )->Write( (pData+4), dwSequence, dwSync );

  // Trash packet or we'll have a massive memory leak
  delete pPacket;
}
```

This replacement library while straightforward to use, is therefore internally quite complicated. Figure 3-13 is a data flow diagram which shows how messages are passed around the networking subsystem. The dashed lines separate the three different thread spaces, circles represent data processes,

closed boxes external entities, and open boxes signify data stores. Messages are indicated by directional arrows.

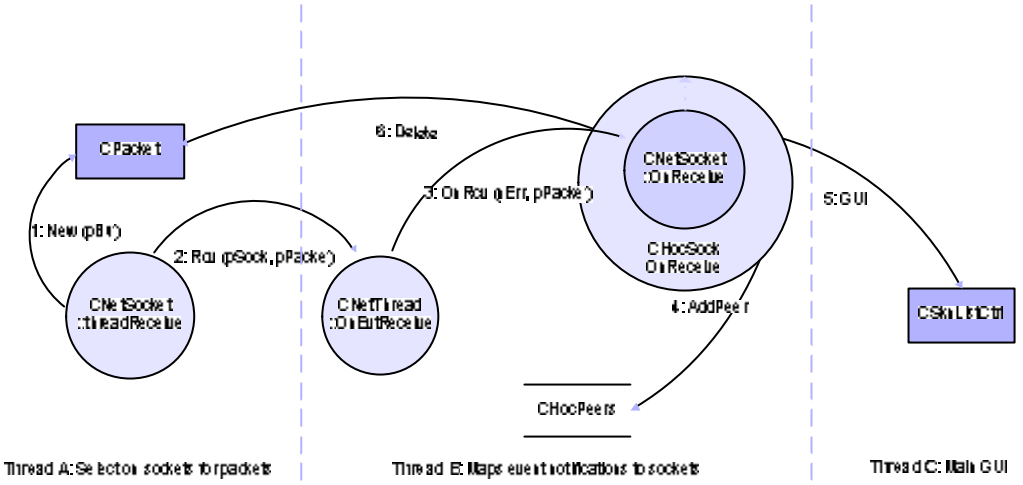


Figure 3-13 Data flow diagram of networking system receiving packet

In this example, Thread A detects that a new packet is waiting on socket X, so it reads the data into a new *CNetPacket* object, and posts a message to the Thread B's queue, telling it that Socket X has received Packet Y, before going back to listening for more incoming packets. A high priority thread must be dedicated to this activity or substantial packet loss will be incurred. Next, Thread B maps the message to a particular instantiation of a *CNetSocket* derived object and calls its *OnReceive* member function. Remember, as this is a virtual function, the subclass will have overridden it, so the correct handler can be dynamically selected at runtime. This event handler is then free to perform whatever processing it wishes on the data received. In this case, the packet is from a peer the system has not encountered before, so it adds the new details to the *CHocPeers* data store, and deletes the *CNetPacket* (asynchronous message sending means that Thread A has no way of knowing when to delete any object it passes on, so it must be done by Thread B instead). Finally, a message is posted Thread C, asking it to update the *CSknListCtrl* that displays the list of nearby peers to the user. Graphical controls should *never* be modified by any other thread, or deadlock may occur.

3.7 Database

tunA's internal database is used for storing a variety of data. First and foremost it functions as a logging mechanism, recording every event that happens in the system, from the user pressing play, to another peer going out of range. Secondly, it is used to store information about any song or person which a user has chosen to 'bookmark'. Finally, it logs the ID3v11 tag of every song that the user listens to (either from their own device or that of a neighbouring peer), to help build up an 'audio portrait' of the music that the user encounters on a day-to-day basis.

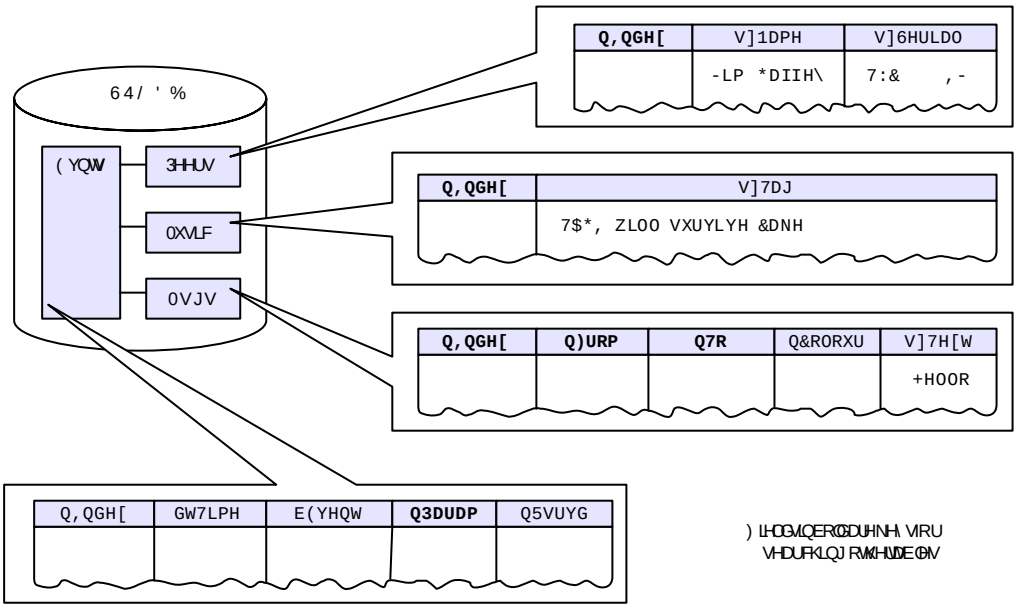


Figure 3-14 Structure of reporting database

The structure of this database is given in the breakouts of Figure 3-14 above. The `Events` table stores one time-stamped record for every system event, and is the main table to which the others are linked. In the example below for instance, when the user sent the text message 'hello' to 'Jim Gaffey', the system created an entry in the `Msgs` table containing a copy of the message, and a corresponding record in the `Events` table noting that a message had been sent. Note: the nParam value in `Events` corresponds to nIndex in `Msgs`, and the nTo in `Msgs`, corresponds to nIndex in `Peers`.

There are several technologies for storing and accessing data on Windows CE. On the storage side, several large database vendors including Microsoft and Oracle provide cut-down versions of their products (i.e. SQL Server and Oracle 9i). Likewise there are several access API's available although some are at best 'quasi-supported'. It was decided to use Pocket Access .CDB files for storage until Windows CE 5.0 is released (yet to ship at time of writing) as until then SQL Server CE is not included in ROM and requires a large install instead. For the interface, after some initial experiments with ADO, a number of C++ classes were written to wrap the WinCE database API instead. This API is quite basic, so it was necessary to write additional routines to generate GUID's, and enforce unique primary keys in tables.

Extract 3-3 – D/B example, performing a SELECT

```
USHORT CDBDatabase::Select( CCeDBDatabase *pTable, CCeDBProp prop )
{
  CEOID       result;        // object id of matching record if found
  USHORT      index = NULL;  // unique_id of matching record, else NULL
  CCeDBRecord record;        // matching record
  int         nRecords = 0;  // num records in table, not recordset

  // Get num records in database table, don't search on empty table
  nRecords = pTable->GetNumRecords();

  if (nRecords > 0 )
  {
    // Seek to first record
    if ( pTable->SeekFirst() == 0 )
      OnError( _T("Select:SeekFirst failed") );

    // Search for record matching prop
    result = pTable->SeekFirstEqual( prop );

    // If match found, return it's unique_id
    if ( result != 0 )
    {
      pTable->ReadCurrRecord( &record );
      index = (record.GetPropFromIdent( PROP_UNIQUE_ID ))->GetUShort();
    }
  }

  // Return unique_id of FIRST matching record, or NULL if none found
  return index;
}
```

3.8 MP3 Playback

3.8.1 Decoding

MP3, or to give it its full title MPEG-1 Layer 3, is a method of compressing audio files. A 'lossy'[24] format, the compression codec depends on two keys principles thresholding, and masking. Thresholding exploits the fact that human hearing is less sensitive in the higher frequency range. This means that higher frequencies can be compressed to a greater extent than lower frequencies. The encoder also reduces temporal redundancies by masking out low amplitude frequencies if a dominant one is present. Using these techniques, compression ratios of 12:1 are achievable without an appreciable reduction in fidelity ('near CD quality' results).

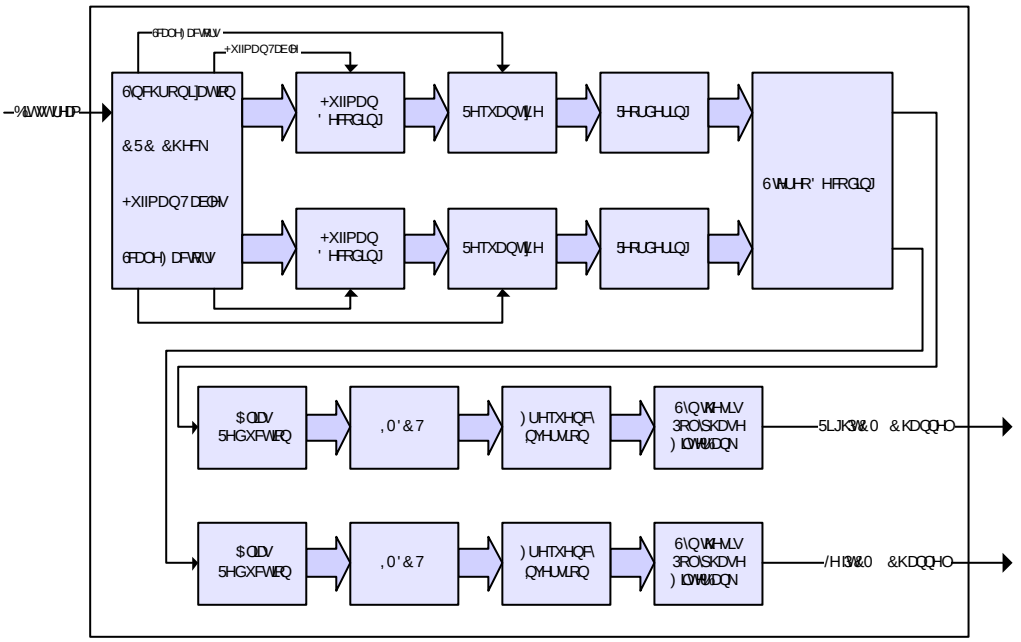


Figure 3-15 Block diagram of the MP3 decoding process

The MP3 algorithms require much more processing power to encode audio, than to decode, which is illustrated in Figure 3-15 above. The audio data is

[24] 'Lossy' compression – the decoded signal is only a close approximation of the original

first Huffman decoded, producing symbols that represent 576 scaled frequency lines. These symbols are then descaled to normal frequency lines, using the scalefactors that are provided in the frame. If the frame contains more than one channel of audio information, the stereo decoder recreates the frequencies for the left and right channels. The last steps produce time samples from the frequency lines by running it through an Inverse Discrete Cosine Transform (IDCT) and a synthesis polyphase filter bank. The innards of these calculations are beyond the scope of this thesis; for a fuller treatment read (Noll, 2000) or (Painter & Spanias, 2000).

3.8.2 File Format

This compressed audio stream is stored as a series of frames, each of which consists of a 32-bit header, about 25ms of audio data, and optionally a 16-bit CRC checksum (Hacker, 2000). As Figure 3-16 below shows, a frame header begins with 11 bits as a sync '11111111111', followed by a number of flags containing enough information to calculate the size of the frame, the level of compression involved, and a few other things such as whether or not the frame contains copyrighted data. Most files also contain 128 bytes of ASCII text at the end in the form of an ID3 tag which contains various descriptive metadata such as the name of the artist, track, album genre and so on (Nilsson, 2001).

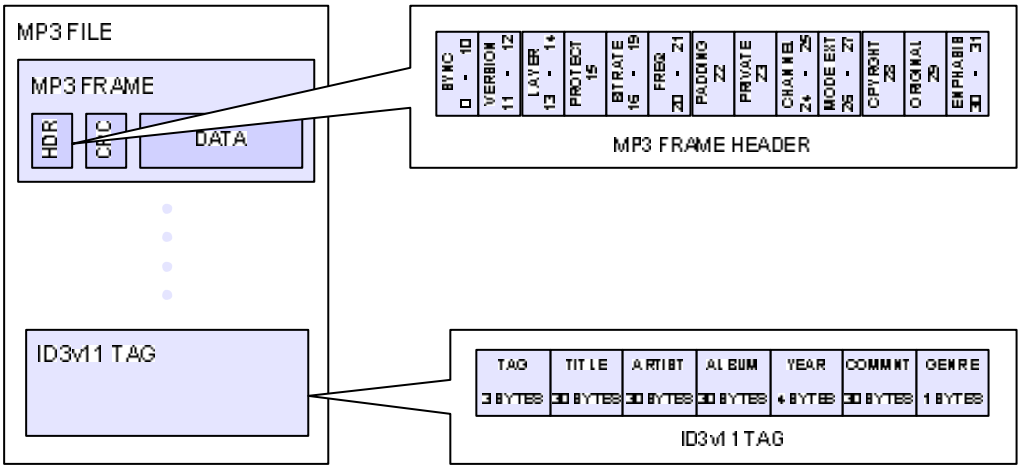


Figure 3-16 MP3 File Format

3.8.3 Streaming

The *tunA* code-base natively supports reading MP3 data, manipulating ID3v11 tags, streaming frames of audio over UDP, and synchronising the status of audio buffers on multiple devices in Ad-Hoc networks. The classes which implement all this behaviour are given in Figure 3-17 below.

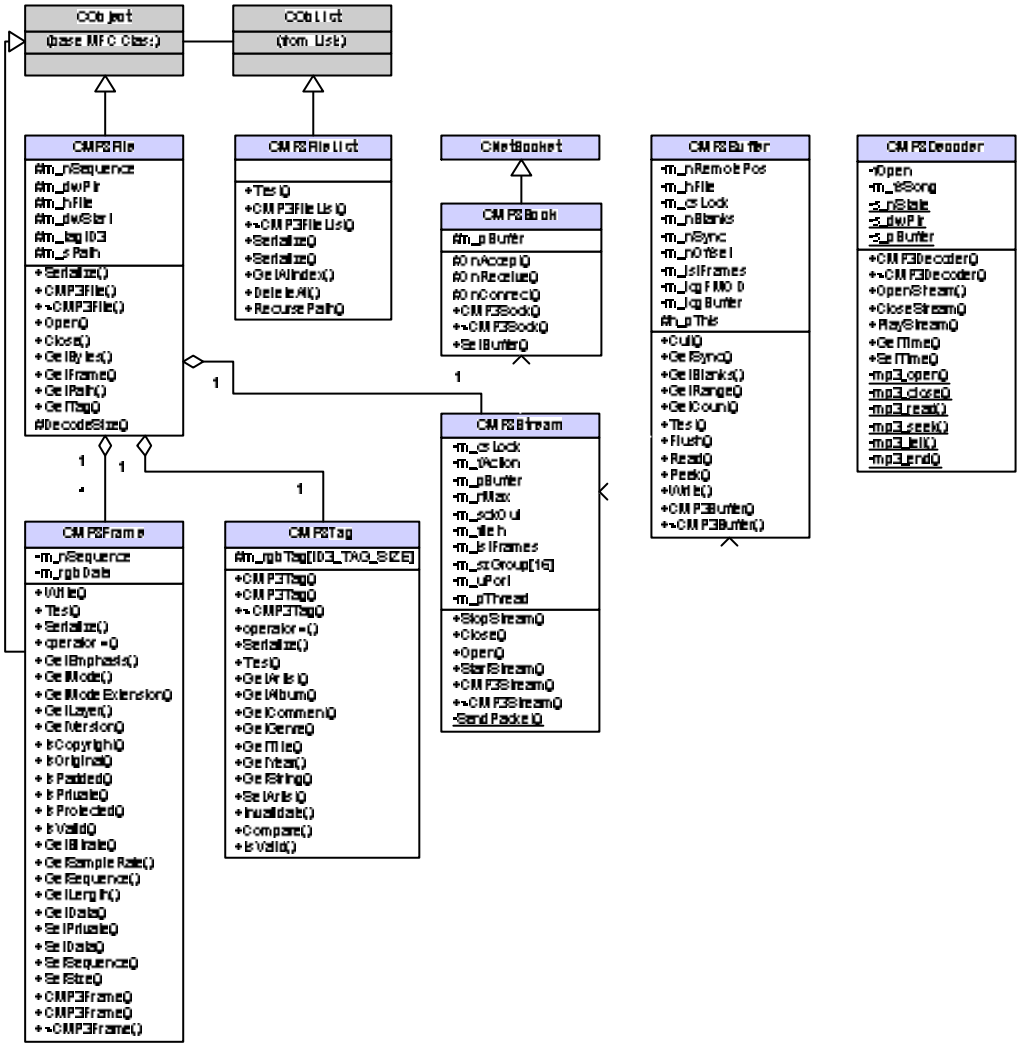


Figure 3-17 UML MP3 Class Diagram

Actual decompressing of an MP3 stream into raw PCM data is however, handled by a separate module. Originally this was a Windows Media control however this closed-source component was too inflexible to be of much use. After that, a version of the original Fraunhofer reference C code was ported to

Windows CE, but proved to be too slow for real-time playback of audio as it is heavily dependent on floating-point math (Pocket PC's do not possess an FPU). Eventually a customised version of the FMOD was used. By over-riding the file call-backs provided by the API, a state-machine based 'virtual file system' was created which reads data from *CMP3Buffer* instead of from a local file whenever FMOD issues a file I/O request, as shown in Figure 3-19:

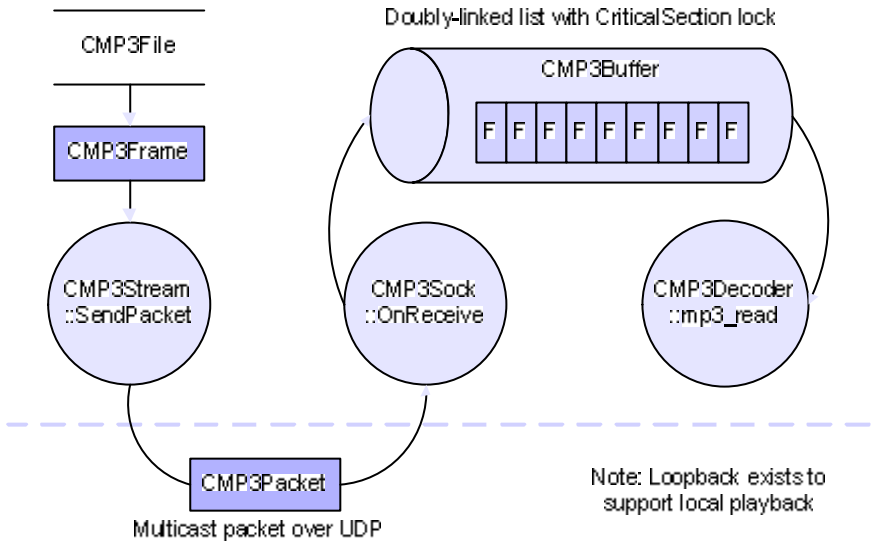


Figure 3-18 Data flow of MP3 streaming

CMP3Stream spawns a thread which reads in ten frames of audio data every 250ms, assigns them sequence numbers, and multicasts them out to neighbouring devices using a *CMP3Sock* object. At the receiving end, another *CMP3Sock* object captures these packets, and adds them to a doubly-linked list contained in *CMP3Buffer*, inserting and removing blank frames as needed, in case a packet is dropped or arrives out of order. *CMP3Decoder* then requests bytes of audio as needed from the list, and passes them to the FMOD audio engine to be decoded into a raw PCM waveform which is played back through a speaker/headphones. This process is the same for local playback as it is for remote streaming, which simplifies the mechanism that keeps buffers on every 'tuned in' peer synchronised, and helps to abstract the audio decoding engine away from the rest of the system.

3.8.4 Synchronisation

The human ear will assume two audio signals are 'coherent' (i.e. from the same source) if they arrive within 30ms of each other. On the Pocket PC platform, this level of synchronisation is extremely difficult to maintain over time due to variances in manufacture (audio crystals), clock skew, OEM dependent timing information, unreliable network protocols, and the lack of a real-time operating system. Despite these obstacles *tunA*'s algorithms are reasonably successful, and should see further improvements if implemented on a dedicated *tunA* device. Synchronisation is achieved in a three-part process, applied for the full duration of the 'shared audio experience', the data for which is included in the header of the packets of MP3 frames that are multicast as part of the audio stream.

- First, a common reference logical clock or 'heartbeat' is established using a derivation of Christian's algorithm (Tanenbaum & Steen, 2002)

- Next, the track position of the remote source is computed using information about the last frame that the decoder requested, and the time it requested it.

- Finally, if the local buffer is determined to be out of sync by more than a pre-determined amount, frames are removed or blanks inserted to bring the local and remote players in line. We could also dynamically adjust the frequency of the local player until the peers matched.

There is ample evidence to suggest that were *tunA* implemented on its own hardware device (see Section 5.3.1) more robust synchronisation would be possible (Li & Rus, 2004)

3.9 Instant Messaging

Instant Messaging has been the subject of much attention lately, and a good deal of work has been performed in the area (Isaacs, Walendowski et al., 2002) (Rodenstein & Donath, 2000). The IM client in *tunA* project however, serves two purposes. First, it allows users of the *tunA* platform to communicate with each other, and secondly, it is used to exchange play lists, messages, and files.

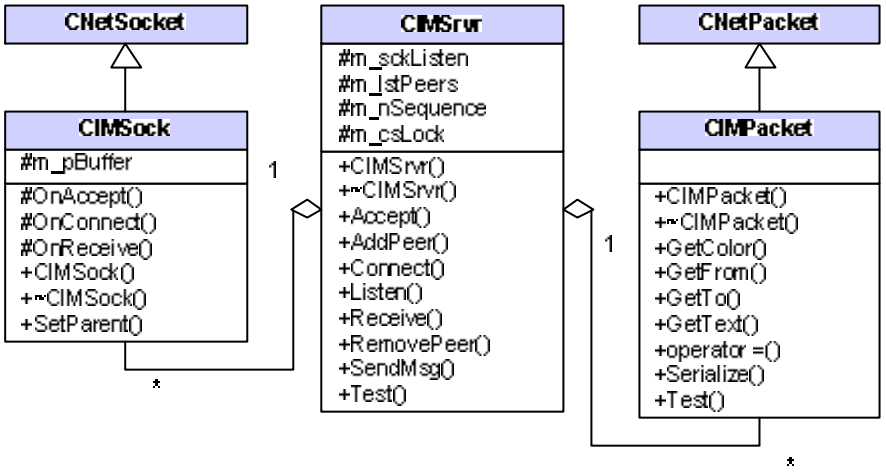


Figure 3-19 UML Instant Messenger Class Diagram

This is achieved by means of a trivial protocol enforced by setting two fields in the *CNetHeader* of Section 3.6, which is the 24-byte header attached to all packets in the *tunA* system. The `type` field is set to either `'TEXT'` `'TBMP'` or `'PROF'`, and the `sequence` number is incremented when sending multi-packet data.

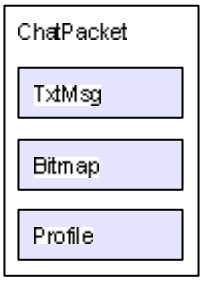


Figure 3-20 IM Packet

Classes in Figure 3-19 implement this, and are responsible for communicating with other peers over standard TCP/IP connections; Section 3.12 deals with the actual input mechanism that captures text input.

3.10 Discovery

tunA uses a straightforward beaconing approach to detect nearby peers, operating at a high level in the networking stack. Lower level details such as IP resolution can be handled in one of two ways: either by configuring each device with the appropriate Ad-Hoc settings (including a statically assigned IP, possibly assigned based on a hash of the device serial number), or by using the Wireless API of Section 3.11, and the built-in *Wireless Zero Config* service. All that is left to the discovery mechanism is to ensure that peers on the same subnet are aware of each other, in this case is by periodically sending out a special 'ping' packet to a pre-assigned multicast group every 1000ms, and maintaining a list of every peer from which a similar packet has been received in the last 3000ms.

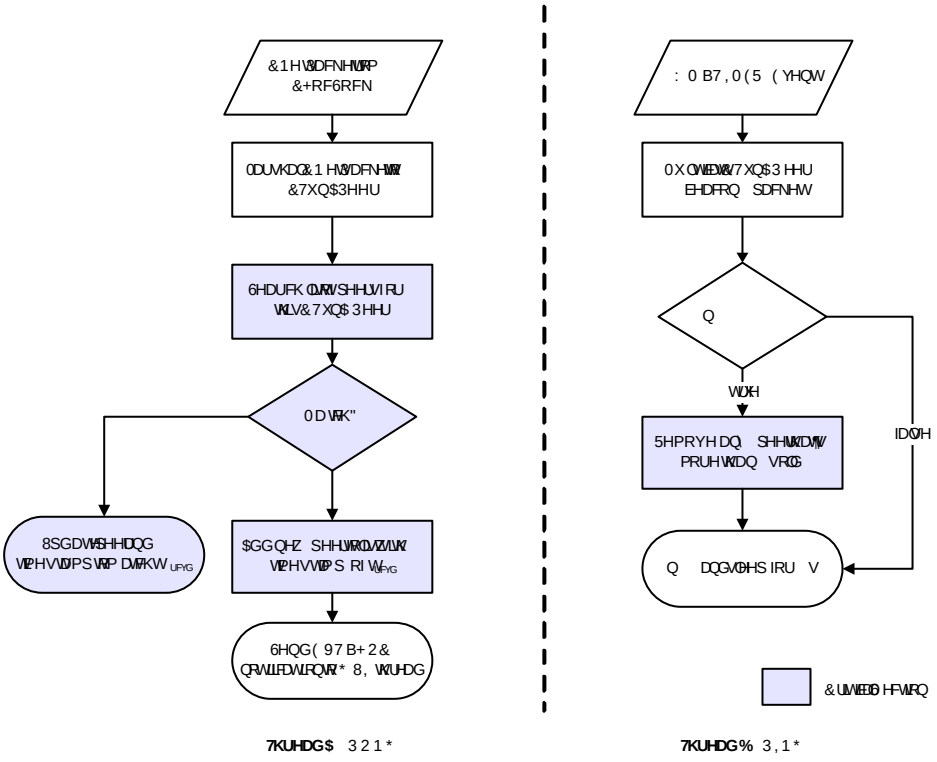


Figure 3-21 Flowchart of Ping/Pong threads

This logic is implemented as two complementary threads, with access to a shared resource (the list of known peers), controlled by a *CCriticalSection*

object, which is represented in Figure 3-21 above by the shaded blue areas. Figure 3-22 is another UML class diagram, this time illustrating the various classes in the discovery system. *CHocSock* subclasses *CNetSocket* for asynchronous socket event notifications and is used to send and receive multicast ping packets to and from other peers. *CHocService* contains two static member functions Ping() and Pong(), which implement the logic described in the flowchart above, and are run as separate threads when the service starts.

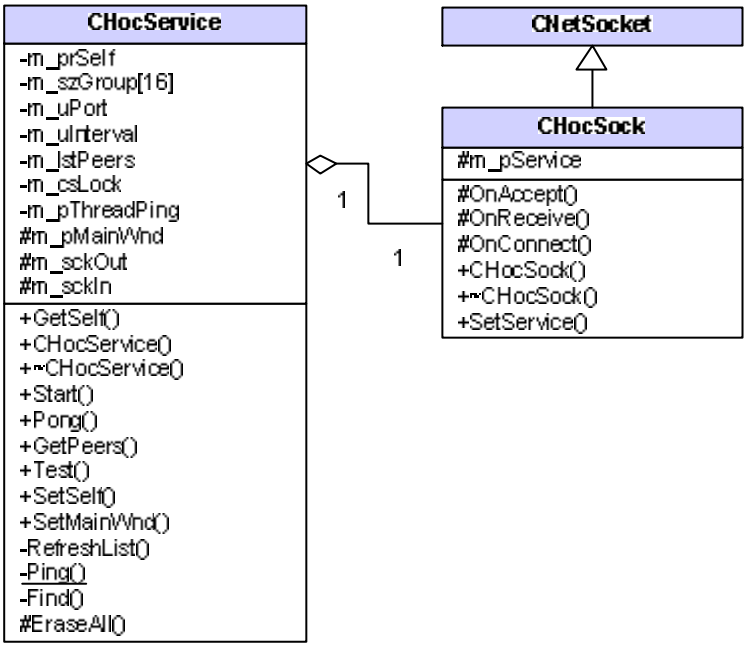


Figure 3-22 UML Discovery Class Diagram

There is one potential drawback to this approach. According to the datasheets for the iPaq h455X range (see Appendix H), in transfer mode the built-in WiFi adaptor consumes 380mA of current, whereas in power saving idle mode, the current drain is only 25mA. The beaconing duty-cycle is quite small – one small packet every 1000ms, but there is insufficient information available about the power-up and power-down periods of the card to determine if this has a significant effect on battery life.

3.11 Utilities

This section details the innards of all the utility classes that perform generally useful 'housekeeping' tasks, such as registry access, dynamic linking of libraries, generating GUID[25]'s, and manipulating hardware. The relationships between these classes are depicted in Figure 3-24 at the end of this section.

A Dynamic Link Library (DLL) is a standard windows file for linking code into an application at runtime. There are two ways basic methods of integrating these libraries into an existing project. The first and most straightforward is to add the corresponding LIB file to the linker settings. A LIB file explicitly lists all the functions exported by the DLL, and their entry-points. Unfortunately this is not always available, as is this case here. The second method is to explicitly list and load each individual function a laborious process, oft prone to error. The *CDllDynamic* class provides a series of macros for dynamically linking DLL's in a more automated fashion:

Extract 3-4 – Dll example

```
// CDllWapi
//
// Wrapper for the Simple CFWAPI dll that allows us to force tunA
// peers to connect to a set SSID, enumerate access points etc.

const int WAPI_MAX_SSID = 32;
const int WAPI_MAX_NDIS_DEVICE_NAME_LEN = 256;

class CDllWAPI : public CDllDynamic
{
public:

  DECLARE_DLL_PROC( int, CALLBACK, Connect, (TCHAR *, TCHAR *,
    TCHAR *, DWORD, DWORD, DWORD, DWORD ) )
  DECLARE_DLL_PROC( int, CALLBACK, Disassociate, (TCHAR *) )
  DECLARE_DLL_PROC( int, CALLBACK, EnumerateAPs, (TCHAR *) )
  DECLARE_DLL_PROC( int, CALLBACK, EnumerateDevices, () )
  DECLARE_DLL_PROC( int, CALLBACK, GetAPCount, (DWORD *) )
  DECLARE_DLL_PROC( int, CALLBACK, GetDeviceCount, (DWORD *) )
```

[25] GUID – Globally unique identifier, a complex number made of various serial numbers, timestamps, random numbers, which is almost certainly guaranteed to be unique

```
    DECLARE_DLL_PROC( int, CALLBACK, GetNextAPData, (TCHAR *, DWORD *,
      DWORD *, BYTE *, BYTE *, BYTE *, BYTE *, BYTE *, BYTE *) )
    DECLARE_DLL_PROC( int, CALLBACK, GetNextDeviceName, (TCHAR *) )
    DECLARE_DLL_PROC( int, CALLBACK, GetSSID, (TCHAR *, TCHAR *) )
    DECLARE_DLL_PROC( int, CALLBACK, SetWEP, (TCHAR *, TCHAR *,
      DWORD *, DWORD *) )

    BEGIN_PROC_TABLE()
      IMPLEMENT_DLL_PROC( int, CALLBACK, Connect, (TCHAR *,
        TCHAR *, TCHAR *, DWORD, DWORD, DWORD, DWORD), _T("Connect") )
      IMPLEMENT_DLL_PROC( int, CALLBACK, Disassociate, (TCHAR *),
        _T("Disassociate") )
      IMPLEMENT_DLL_PROC( int, CALLBACK, EnumerateAPs, (TCHAR *),
        _T("EnumerateAPs") )
      IMPLEMENT_DLL_PROC( int, CALLBACK, EnumerateDevices, (),
        _T("EnumerateDevices") )
      IMPLEMENT_DLL_PROC( int, CALLBACK, GetAPCount, (DWORD *),
        _T("GetAPCount") )
      IMPLEMENT_DLL_PROC( int, CALLBACK, GetDeviceCount, (DWORD *),
        _T("GetDeviceCount") )
      IMPLEMENT_DLL_PROC( int, CALLBACK, GetNextAPData, (TCHAR *,
        DWORD *, DWORD *, BYTE *, BYTE *, BYTE *, BYTE *, BYTE *,
        BYTE *), _T("GetNextAPData") )
      IMPLEMENT_DLL_PROC( int, CALLBACK, GetNextDeviceName, (TCHAR *),
        _T("GetNextDeviceName") )
      IMPLEMENT_DLL_PROC( int, CALLBACK, GetSSID, (TCHAR *, TCHAR *),
        _T("GetSSID") )
      IMPLEMENT_DLL_PROC( int, CALLBACK, SetWEP, (TCHAR *, TCHAR *,
        DWORD *, DWORD *), _T("SetWEP") )
  END_PROC_TABLE()
};
```

Extract 3-4 above shows an example of this in action. More specifically it shows how *CDllWapi* can be defined and loaded in only a few lines. This class is used to enumerate the various network adaptors installed in a system, reconfigure network settings, disassociate from any infrastructure points, force a card into Ad-Hoc mode, and change the current SSID. Finally, *CDllIpaqUtils*, provides functions to activate and deactivate the networking hardware present in certain iPaqs.

Other useful sets of tools are stored in *CVORegistry*, which is used to get, set, and edit keys and values in the WinCE registry (a central data store that the operating system and applications are encouraged to store configuration data in, rather than maintaining many individual settings files). *CSTUtil* and

CSTComError are a library of miscellaneous routines, one of the more useful being *CoCreateGuid()* which returns a GUID when called.

As Section 3-8 details, Windows CE is not a real-time operating system by any dint of the imagination, which makes having accurate timing information even more important. *CCeTimer* then is a C++ class which wraps up the three basic API calls for retrieving timing information from the O/S:

- *QueryPerformanceCounter()* – Value of high-resolution OEM counter
- *GetTickCount()* – Time in ms since WinCE was started
- *GetSystemTime()* – The current system time and date

Unfortunately, the high-resolution is entirely OEM dependent, and may not even be installed. The other functions return *may* return a value in milliseconds, but are not guaranteed to be anywhere near as accurate, and in practice are not. In addition, there is no means for measuring the accuracy of the audio crystal, or it's skew over time. These caveats make it very difficult to develop time-sensitive applications on the PocketPC platform, and as a rough gauge of these variances across multiple devices, the *ClockTst* depicted stub in Figure 3-23 was developed.

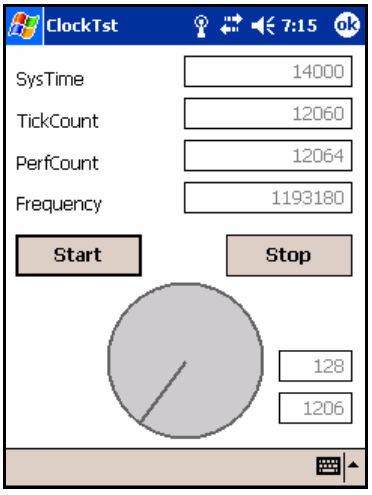


Figure 3-23 ClockTst, a stub application to test WinCE timing data

Finally in this section, *CSkinFile* and *CTextFile* provide support for file I/O operations. *CTextFile* wraps the Win32 API functions for opening, closing, reading/writing, and seeking in a file, and also adds support for the Unicode file format. WinCE uses Unicode strings natively for most operations, with the exception of certain library calls (WinSock being a prime example). Unicode differs from the old ANSI standard, in that it uses 16-bits instead of 8 to represent characters, allowing for a greatly expanded alphabet. Text files encoded in this format have a header of 'FF FE'.

	ANSI	Unicode
// // AriSkin.skn – Default skn // [MetaData] SkinName = AriSkin Description = Default skin Author = Julian Moore Date = 18 Jan 2004 Version = 1.8 [Controls] PlayButton = 50, 110, play.GIF StopButton = 50, 150, stop.GIF NextButton = 50, 180, next.GIF PrevButton = 50, 210, prev.GIF . . .	2F 2F 0D 0A 2F 2F 20 41 72 69 53 6B 69 6E 2E 73 6B 6E 20 2D 20 44 65 66 61 75 6C 74 20 73 6B 6E 0D 0A 2F 2F 0D 0A 0D 0A 5B 4D 65 74 61 44 61 74 61 5D 0D 0A 53 6B 69 6E 4E 61 6D 65 20 20 20 20 20 3D 20 41 72 69 53 6B 69 6E 0D 0A 44 65 73 63 72 69 70 74 69 6F 6E 20 20 3D 20 44 65 66 61 75 . . .	**FF FE** 2F 00 2F 00 0D 00 0A 00 2F 00 2F 00 20 00 41 00 72 00 69 00 53 00 6B 00 69 00 6E 00 2E 00 73 00 6B 00 6E 00 20 00 2D 00 20 00 44 00 65 00 66 00 61 00 75 00 6C 00 74 00 20 00 73 00 6B 00 6E 00 0D 00 0A 00 2F 00 2F 00 0D 00 0A 00 0D 00 0A 00 5B 00 4D 00 65 00 74 00 61 00 44 00 61 00 . . .

Table 3-5 – Comparison of ANSI and Unicode file formats

CSkinFile subclasses *CTextFile* and adds additional methods for parsing the .SKN files, which enables the look-and-feel of the user interface to be customised easily (see Section 3.5). An extract from one such file is given in Table 3-5 above, complete with a hex dump of it encoded in both formats.

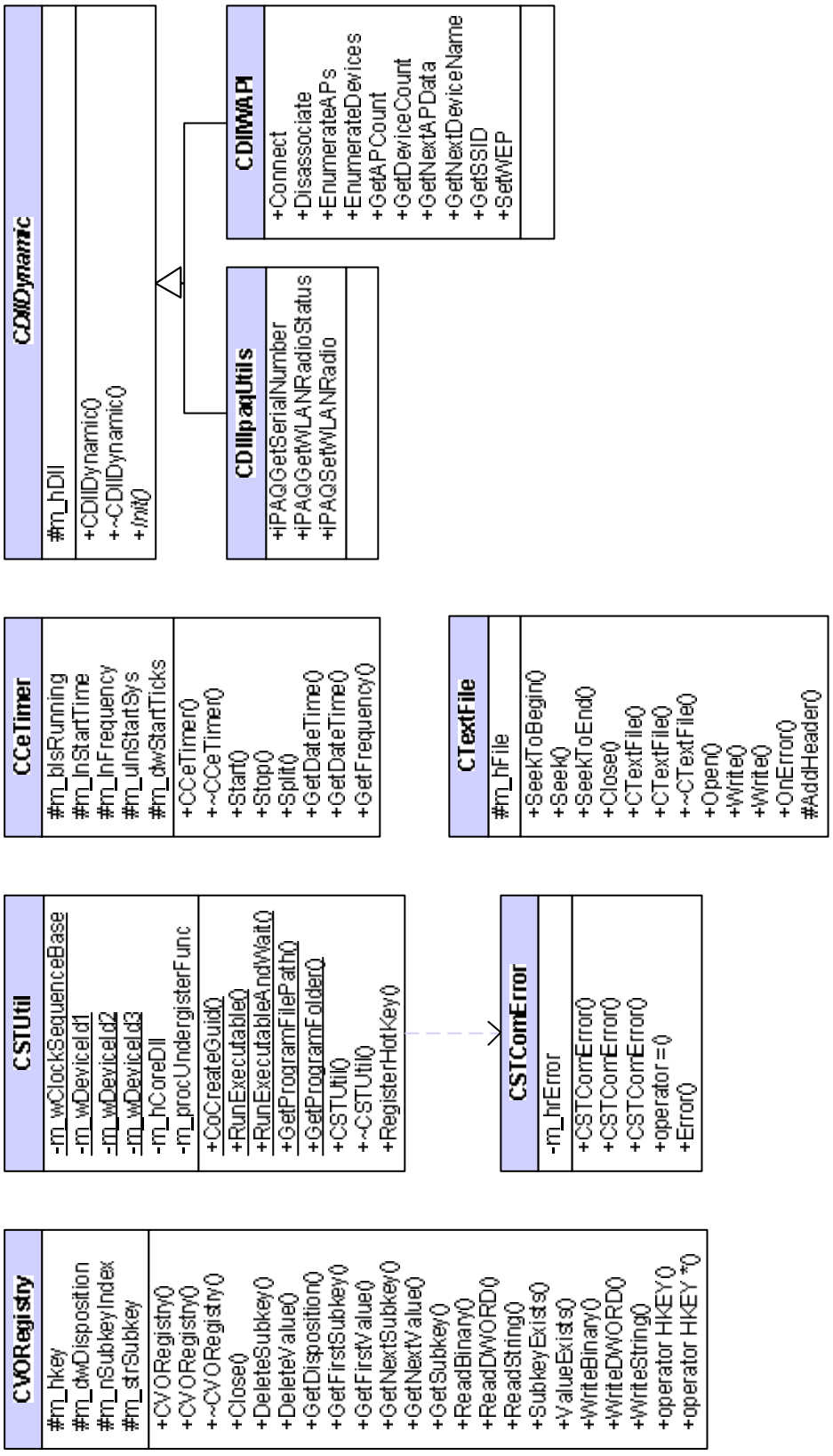


Figure 3-24 UML Utilities Class Diagram

3.12 T9 Text

T9 is the commercial name for a breed of predictive text entry systems. While there are many subtle variations, all of them attempt to minimize the number of button presses required to enter a word on a device that has a reduced number of keys, such as a mobile phone. In keeping with the 'Hi-Fidelity' design goal from chapter two (innovation in one area should not reduce the user experience in another) the text input system (see Section 3.5) was augmented with T9-style logic, the overall flow of which is shown below in Figure 3-25.

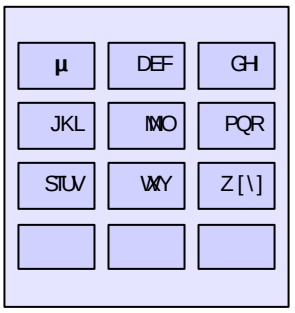
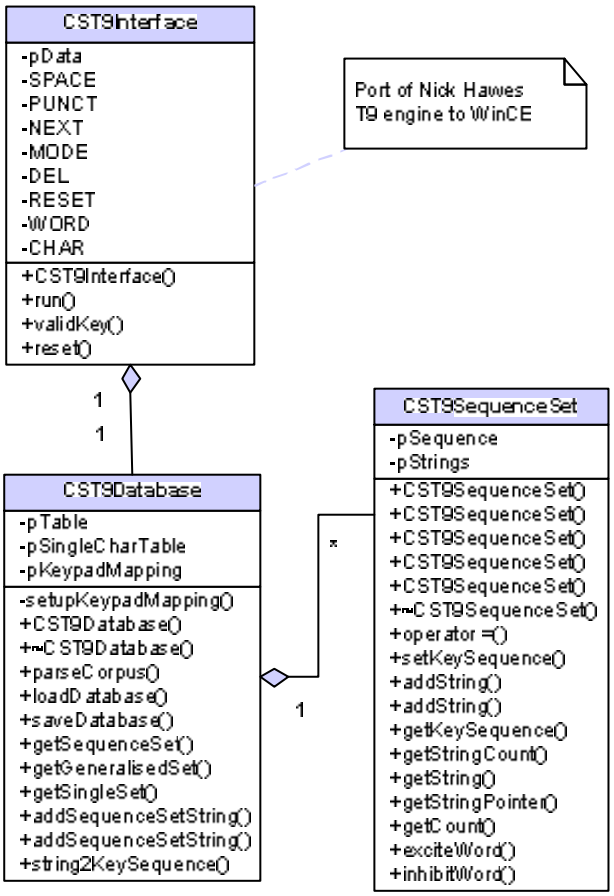


Figure 3-26 Keypad

In a predictive text system, the user instead taps each key only once (e.g. to select c, a user enters '2', not '222' as above). Using this technique, to enter the word 'cop' requires only three key presses '267'. This of course introduces confusion, as '267' could also be interpreted as 'qmp', 'bnp', or 'bms'. To resolve these ambiguities, a large dictionary of words and their key combinations is used. This can also be weighted so that in the case of there being more than one valid word for any given key combination, the most likely candidate is offered first.



3.13 Summary Remarks

This chapter dealt with the primary contribution of this work, a functioning software prototype of a socialising walkman. The design and implementation of *tunA*, were broken down by core functionality: main application, user interface, networking library, database logging, media playback / streaming / synchronisation, instant messaging, peer discovery, utility classes, and T9 predictive text input. For each of these categories multiple technical hurdles were identified, and the approach taken to overcome them was described using a combination of high-level flow charts / data flow diagrams / sequence diagrams, and lower-level code extracts, and class descriptions. More in-depth information if required is available in the cited texts/papers, and in some cases in source-code form in the Appendices, or on the companion DVD-ROM.

The next chapter offers some analysis of several early evaluations of the current prototype, as well as plans for a larger study which has yet to be completed at time of writing. Brief details of several other live installations, exhibitions and presentations of the *tunA* project are included at the end of the chapter.

4 Evaluation / Results

4.1 Ethnographic Studies

The *tunA* project is a multidisciplinary effort, a technical solution to a social research problem. This chapter summarizes the research methodologies and results of a number of user studies that were carried out at Media Lab Europe (MLE) and at the National College of Art and Design (NCAD) into the potential uses of this software platform.

This section opens with the results of an initial questionnaire, which focused on musical tastes, social networks, and familiarity with technology. From these respondents, several volunteers were selected for a second study which involved an evaluation of the user interface, a small-scale deployment of the system on a college campus, and pre and post interviews with all participants. A much larger third study involving a hundred users over the course of several weeks is being prepared in co-ordination with a large US university, and early plans for this are also discussed briefly. Finally, details are given of several other live exhibitions and installations of *tunA* which were not part of a formal study, but from which valuable feedback was nonetheless gained.

Copies of the questionnaires, and transcripts of all the interviews are provided in Appendices F and G respectively; the datasets for all graphs are supplied in Excel format on the attached DVD-ROM.

4.2 Study One: Initial Questionnaire

The first study consisted of a five page questionnaire which was distributed to a large number of volunteers as preliminary research into the musical tastes, social networks, and technological habits of *tunA*'s intended audience. This section presents the methodology employed, and results obtained. Summary remarks are left until the end of the chapter.

4.2.1 Methodology

A total of 76 individuals took part in the initial survey, which was conducted at NCAD and MLE. The majority of the respondents were in their early twenties, and from an arts background. Roughly equal numbers of males and females participated. The questionnaire was divided into three main sections, each of which was intended to confirm some initial assumptions (e.g. people enjoy music, are eager to connect to others etc...), establish some baselines (e.g. what kind of personal data are the public comfortable sharing with others), and gauge the popular appeal of the main ideas behind *tunA*.

The majority of questions were based around a seven-point Likert[26] scale, with the frequency distribution of responses being averaged to obtain a mean such that:

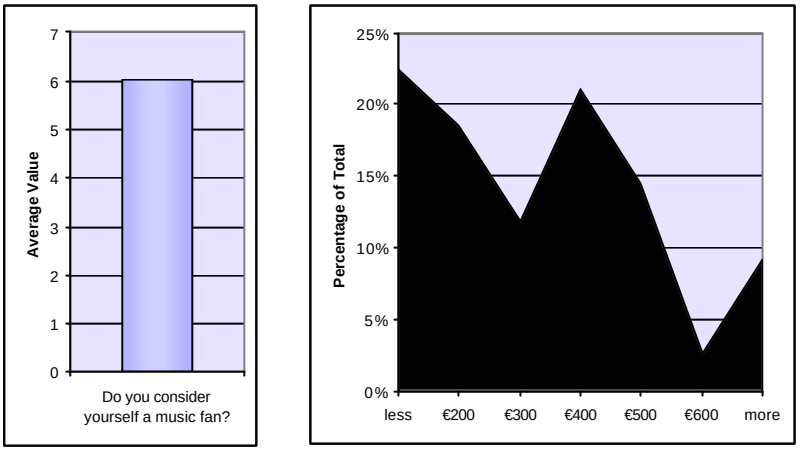
$$mean\ \bar{x} = \frac{\sum xf}{\sum f}$$

In each case, the number of individual responses n_x is given in the left-hand column; where the question called for more than one answer, the nomenclature t_x is substituted. Also, for aesthetic reasons abbreviations have been used when titling some of the graphs (e.g. P2P instead of 'peer-to-peer'). An excel spreadsheet containing all the raw data on which this section is based, is supplied on the companion DVD-ROM.

[26] Likert – Gradient scale, for example: 'strongly disagree', 'disagree' 'somewhat disagree', 'undecided', 'somewhat agree', 'agree', 'strongly disagree'

4.2.2 Analysis – Music

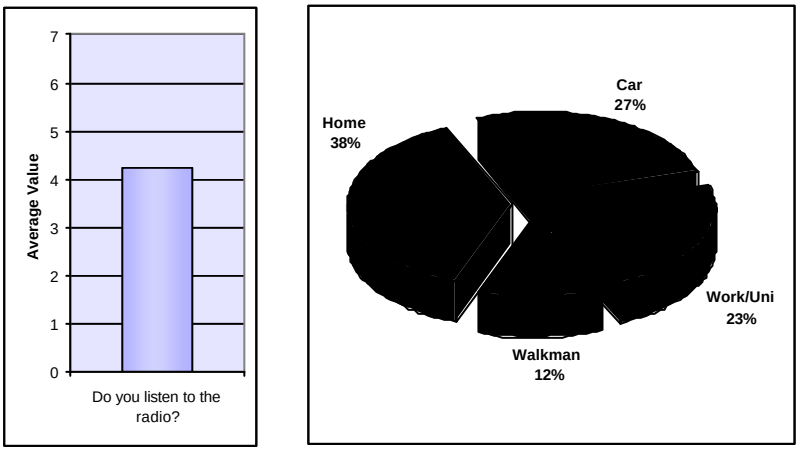
For a common activity to be successful in fostering social bonds, it seems reasonable to suggest that it must evoke a certain level of response from everyone involved. To test the assumption that music is one such medium, participants were asked to rate their interest in it out of seven, and estimate their average annual investment in albums. As Graph 4-1 shows, almost all participants perceived themselves to be big music fans, with 47% of them spending €400 or more a year on music.



$n_1 = 76$
$n_2 = 76$

Graph 4-1 Investment in music per annum

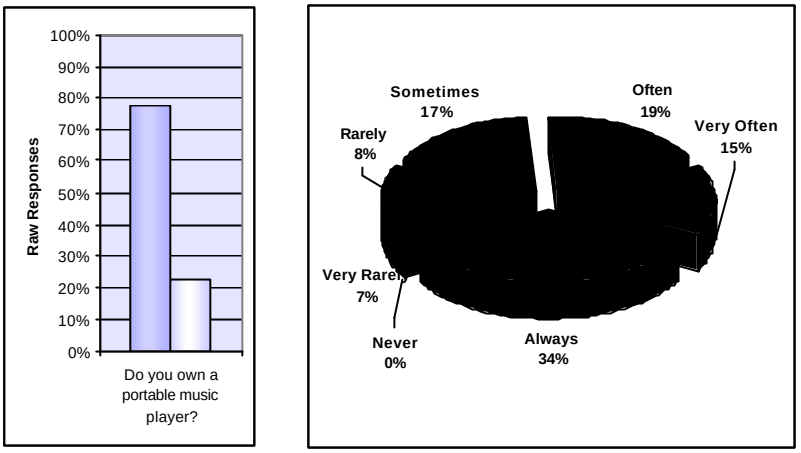
Next, participants were asked how often they listening to the radio, and where they found themselves doing this most often (Graph 4-2).



$n_1 = 76$
$t_1 = 105$

Graph 4-2 Greatest exposure to radio

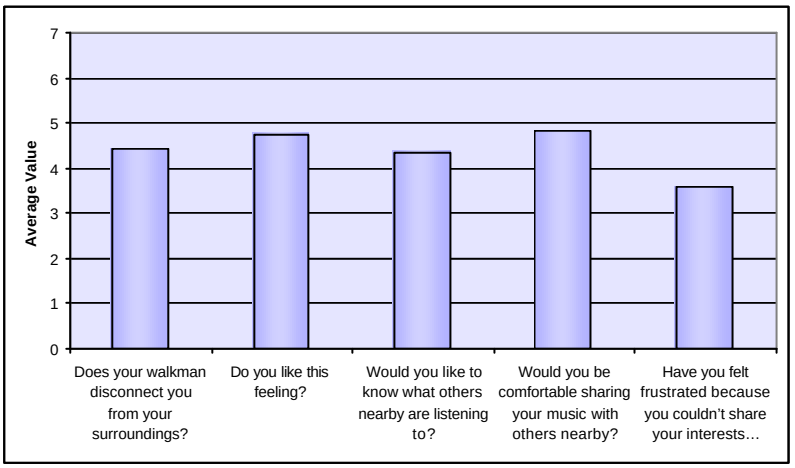
Interestingly, a majority claimed to 'tune in' while in relatively solitary environments such as at home or in the car.



$n_1 = 76$
$t_2 = 59$

Graph 4-3 Walkman ownership and frequency of use

As a large critical mass of users would be required in an uncontrolled environment for *tunA* to be useful, it is reassuring to note then that the penetration of portable music devices is still high (Graph 4-3), with 78% of respondents claiming ownership of one. Additionally, 68% of these users claim to listen to them *often*, *very often*, or *always*.



$n_1 = 67$
$n_2 = 58$
$n_3 = 73$
$n_4 = 67$
$n_5 = 72$

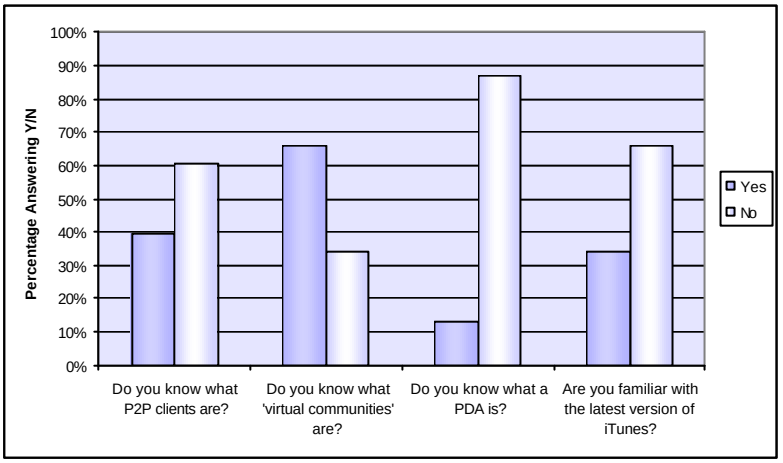
Graph 4-4 Reaction to *tunA* concept

Finally in this area, the beliefs behind *tunA* were introduced (Graph 4-4) with somewhat mixed results. While the majority felt that their walkman was

disconnecting them from their environment (mean of 4.41 / 7.0), they also appeared to enjoy this feeling, which is understandable enough if it spares them the distracting racket of the urban soundscape. On the other hand, the idea of being able to tune into other people, and the requirement to shares ones own music both received strong support. A surprisingly large number of people even went so far as to say that they have 'felt frustrated by not being able to share their musical interests with others around them'.

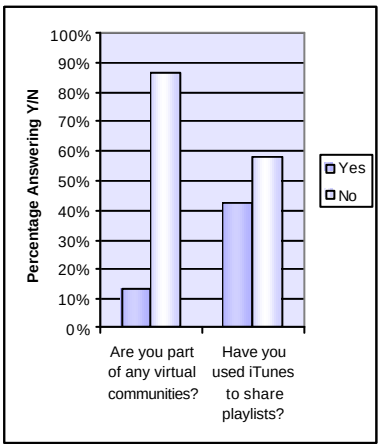
4.2.3 Analysis – Technology

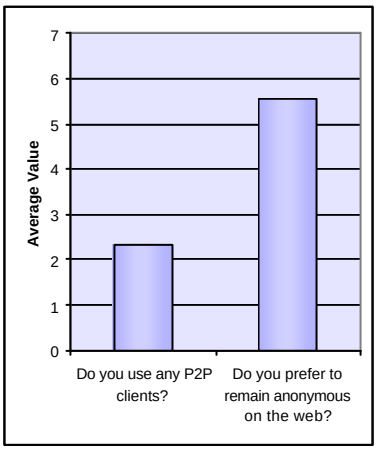
The second section of the questionnaire covers participants' familiarity with, and use of technology. As evidenced in the *Slashdot* coverage (Appendix D) highly technical users often have different expectations from software.



$n_1 = 76$
$n_2 = 76$
$n_3 = 76$
$n_4 = 76$

Graph 4-5 Familiarity with new technology

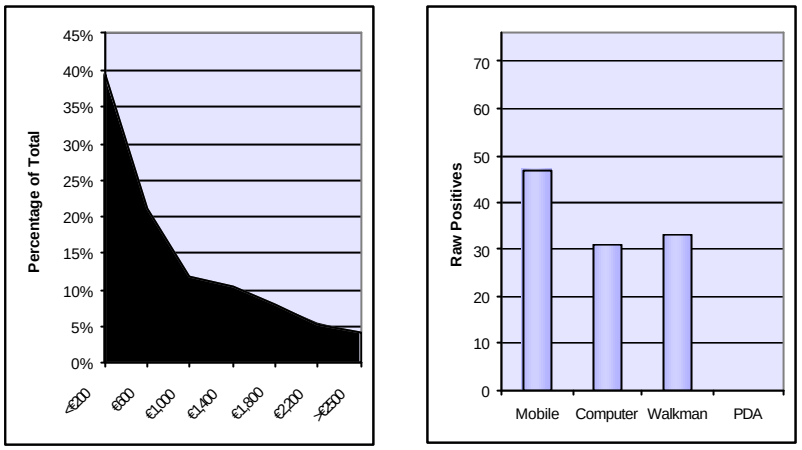




$n_1 = 76$
$n_2 = 26$
$n_3 = 76$
$n_4 = 33$

Graph 4-6 Internet habits and trends

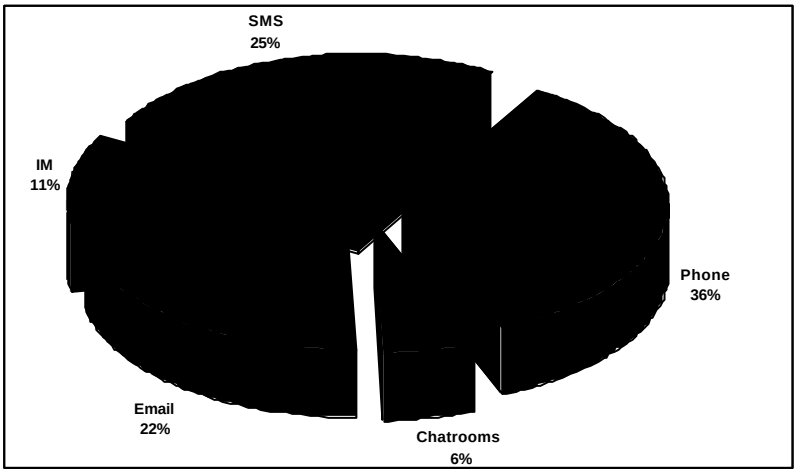
Graphs 4-5 and 4-6 above show that the sample group had a fairly mixed grasp of the latest developments in this area, with relatively few having heard of either PDA's or P2P clients (see Section 2.4.2). Despite many having heard of virtual communities (blogs, mail-lists, newsgroups) very few (about 10%) claimed membership of any. Of greater surprise, given that 54% of participants claimed to be *very strong* fans of music, is the small number claiming to use file-sharing software.



$n_1 = 76$
$t_1 = 111$

Graph 4-7 Investment in technology per annum

The bias introduced by the fact the majority of participants were art students, is particularly evident in Graph 4-7, which shows that 40% of them spend less than €200 per annum on technology, most of which is on mobile phones.



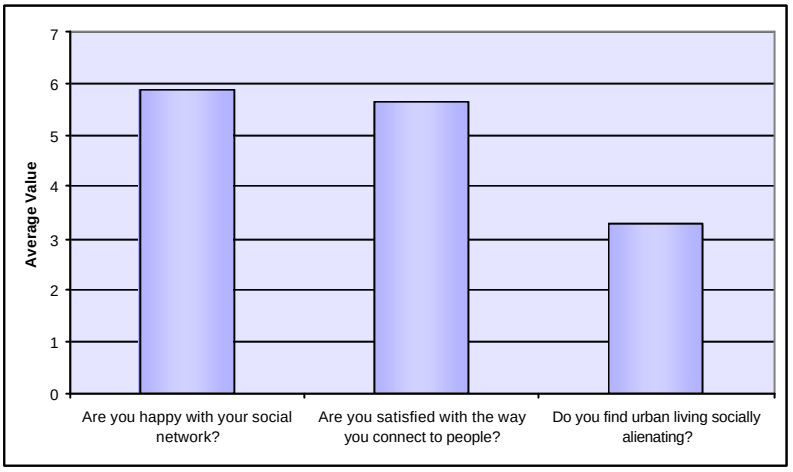
$t_1 = 1033$

Graph 4-8 Preferred means of connecting to people

Finally volunteers were asked to put several popular means of communication in their order of preference for communicating with others; their answers were averaged and the results presented as a pie-chart in Graph 4-8 above. Internet chat-rooms seemed particularly unpopular with only 6% of the overall votes, while the huge popularity of SMS and IM (36% combined) provided further incentive to add an instant messenger to *tunA*.

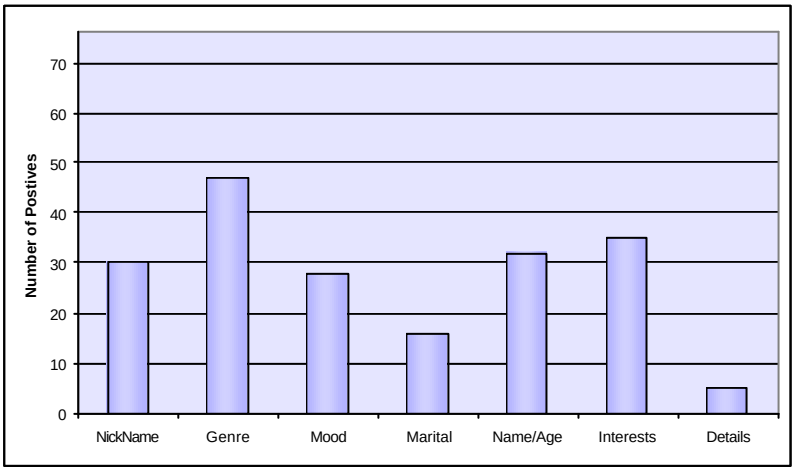
4.2.4 Analysis – Social Networks

The final section asked participants about their social networks. Again, as it was from them that volunteers for a more in-depth study would be drawn, it was necessary to determine a few baseline measurements.



$n_1 = 75$
$n_2 = 76$
$n_3 = 73$

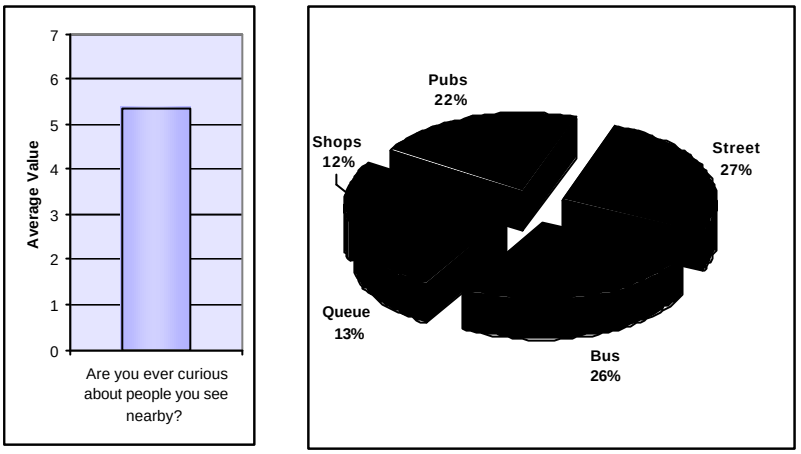
Graph 4-9 Urban life from a social perspective



$n_1 = 76$
$t_1 = 193$

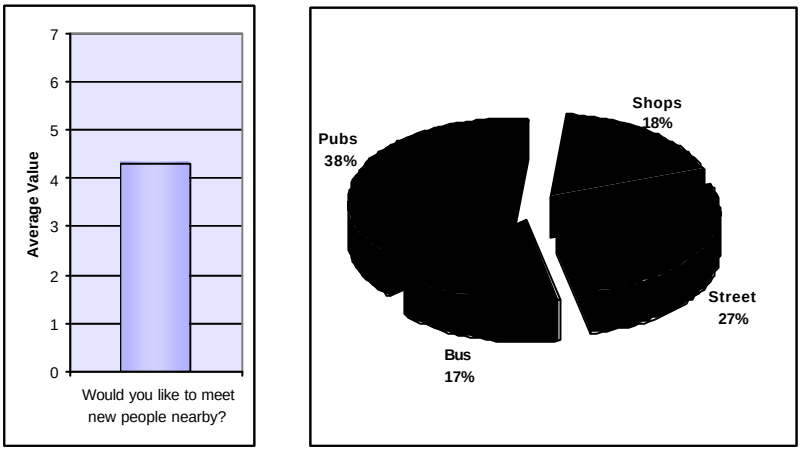
Graph 4-10 Personal info that can be comfortably shared

A large majority of respondents reported that they were *very happy* their social networks, and the way in which they connect to people. This however is probably not reflective of society at large, as students are already members of a large social institution. When asked what types of information they would feel comfortable sharing with strangers, participants seemed quite coy. This influenced the design of *tunA*; very little personal info is revealed by default – it is expected that users will expose this via the IM instead.



$n_1 = 75$
$t_1 = 170$

Graph 4-11 Areas where most curious about strangers



$n_1 = 74$
$t_1 = 101$

Graph 4-12 Inclination to befriend nearby strangers

Finally Graphs 4-11/12 show that while people are most curious about others when on the bus or street, they prefer to meet in bars, reinforcing the notion of the pub as being central to Irish social life. There is also an unusual drop (5.36 to 4.31) between being interested in others nearby, and wanting to meet them.

4.3 Study Two: Small MLE / NCAD Trials

Six volunteers were selected for a second set of trials, which involved an evaluation of the user interface, a small-scale deployment of the system on a college campus, and pre and post interviews with each participant.

4.3.1 Methodology

Four males and two females took part, and were quizzed in greater detail about the subject areas of the initial questionnaires, and their responses to them. These semi-formal interviews were videotaped on the day preceding the field study, as were the interface evaluations in which they were asked to perform a number of tasks using the *tunA* software. Their responses to these tasks were recorded and timed, and each user completed a short debrief questionnaire after the test. The following day each participant was given an iPaq, and a charger (the trials were conducted over roughly six or seven hours, which is about twice the battery life of the units). For most of the day the participants were asked to go about their normal business, using the *tunA* devices in lieu of their walkmans, as it was hoped (with some justification) that the small size of the NCAD campus where the trials were conducted would mean that they periodically encountered one another. To ensure that at least some interaction took place, each user agreed to meet in the canteen during the lunchtime period, where a MediaLab researcher (Arianna Bassoli) was on hand to monitor their behaviour first-hand. As a final measure, a round of semi-structured, post-trial interviews was taped at the end of the day.

4.3.2 UI Evaluation

While the group may have small, it is said that a sample of five or more users is sufficient to expose out the major flaws in a GUI design (Nielsen, 1994). As part of this process, users were asked to perform a series of tasks, the response times to which were measured and are presented in Table 4-1 below. Unfortunately, poor lighting conditions in the MediaLab where the evaluation

took place, meant that the video footage on which this table is based is grainy and out-of-focus in places, resulting in an incomplete set of figures.

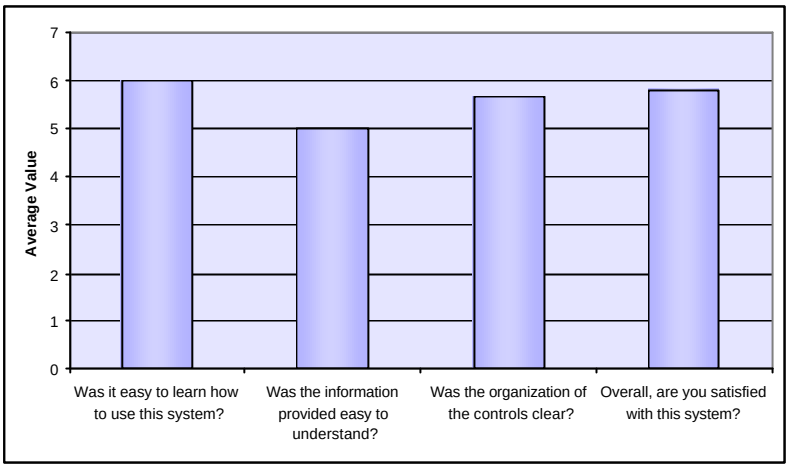
GUI Evaluation Results							
		Alan	Chris	Dara	James	Lisa	Paula
	LISTENING TO YOUR OWN MUSIC						
1	Navigate to your playlist		37				
2	Select a song from your playlist						
3	Play the song	4	4	7	40	11	17
4	Stop the song	2	1	4	2	2	2
5	Select and play another song	4	6	6			4
	INTERACTING WITH OTHERS						
6	Navigate to the list of other users		50				45
7	Select a user	15				17	
8	"Tune In" to this user	30	20	18	105	35	22
9	End the connection	3	40	3	6		
10	Add this song to your favourites	18	73	16	4	44	70
12	Chat with this user	7	5		4		
	INSTANT MESSAGING						
13	Type the short msg "hello"						
14	Send this msg to the other user	35	95	76	70	30	42
15	Navigate to your favourites			3	26		

Table 4-1 – GUI Evaluation Results

As an added complication, the presence of a large number of outliers (e.g. it took 'James' 40 seconds to discover how to play a track, as opposed to times of 4, 4, and 7 seconds from 'Alan', 'Chris', and 'Dara') severely reduced the value of this data, and it was necessary instead to rely on the results of the debrief questionnaire to determine users level of comfort with the interface.

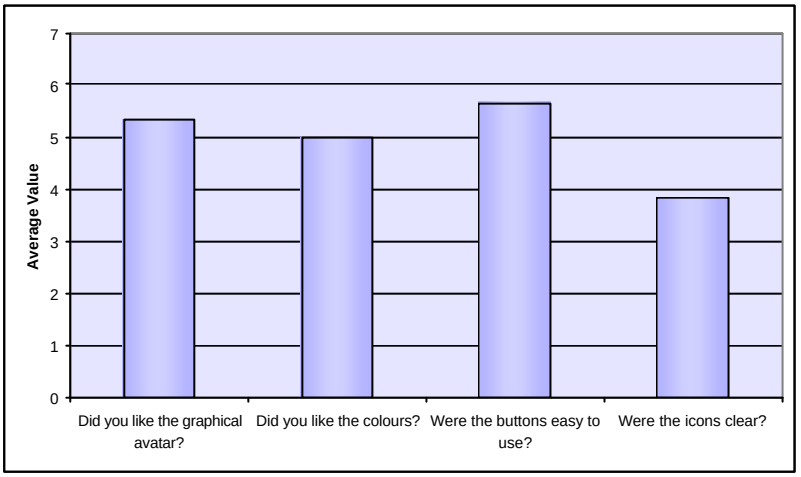
4.3.3 Debrief Questionnaire

By default *tunA's* GUI (as seen in Figure 3-7) is split into four tabbed screens, divided by functionality. The decision to do this would seem to have been a good one, as it was received very favourably during this pilot study, as shown in the graphs below. The only criticism levelled at the UI, was that the device itself felt expensive and frail (particularly the touch-screen).



n=6

Graph 4-13 Layout of tunA's interface

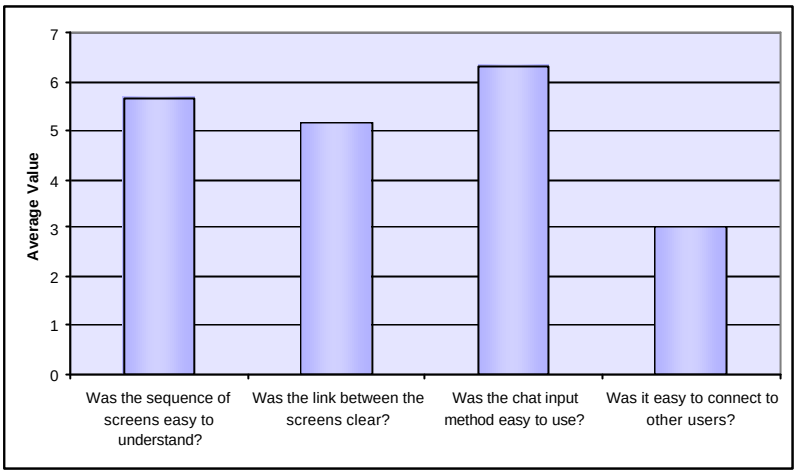


n=6

Graph 4-14 Aesthetic appeal of the UI

Overall it would appear from these that users found the software easy to learn, simple to use, and were very satisfied with the system. Some of the icons used in the trial version of the skin were confusing to some users, were subsequently replaced. In addition the 'link between the screens' was not abundantly clear, but in retrospect this may have been a misleading question, as the tabs were designed to be quite self-contained (e.g. the playback controls, avatar, and playlist are all on the same screen). Finally, as Graph 4-15 below shows, the users did have difficulty in connecting to others – although this may well have been to a problem with the networking core of the software which was still

being refined at the time of the trial, more than a problem with the layout of onscreen controls themselves.



n=6

Graph 4-15 Measuring ease of use

4.3.4 Interviews

The semi-formal interviews provided a large amount of qualitative data, a full analysis of which however, is beyond the scope of this thesis. One surprising result of this information however, was the enthusiasm with which the idea of a 'socialising walkman' was received, as illustrated in the following extract:

> *Q: So do you think it did help establishing connections?*
>
> *A: Definitely yeah! Cause I mean you can just…have a laugh…well I think people are just social creatures and when people share an experience…they automatically bond doing that…you can't help but…so even with a stranger listening to the same music…like with Paula, I didn't know her at all, but we were hanging out as we knew each other, I'm not sure how many times I asked her name, it didn't really matter, we didn't care, we were just hanging out…and you wouldn't get that otherwise…so it made a massive difference, it flipped the whole social thing completely, it's a much more relaxed/ enjoyable kind of thing, it makes you skip all kind of b*****it of "hey how are you?". I thought it worked well with both good friends and strangers…very different but good…*

The other interviews continue in a similar vein. The curious can find transcripts of them in Appendix G.

4.4 Study Three: Large American Field Study

The high cost per unit of the hardware devices (approximately €650 for an iPaq, extended battery, headphones, and memory card) meant that earlier user studies were fairly small-scale, with no more than six users at a time. However, a tentative partnership has been forged with a large US university that is interested in running an extended joint-trial, and has over a hundred iPaqs available for such use. Unfortunately, at the time of writing this study has yet to be conducted; some early plans are offered nonetheless.

4.4.1 Objectives

While the smaller pilot studies were useful for evaluating the user interface, gauging public reaction to the idea of a socialising walkman, and testing various releases of the software build – cost and time constraints meant they were unable to explore the longer-term goals of the overall *tunA* project – to uncover new forms of social interaction and music distribution. It should be noted that these broader goals are separate and distinct from the focus of this thesis which is concerned only with the development of the *tunA* software. It is hoped that the next round of studies will take place over a longer period of time (several weeks) and involve around a hundred subjects. A study of this scale would generate huge volumes of data, from which both qualitative and quantitative answers to the ethnographic questions posed in the first chapter could be drawn. In review, these were:

- **Listening habits** – Is the knowledge that others may be aware of what one is listening to, sufficient to alter to a person's choice of music? To what extent, and what can be done to mitigate this?
- **Evolution of tastes** – How do musical tastes evolve? Do the musical preferences of our peers exert a significant influence on our own?
- **Audio portraits** – By cross-referencing location data with a record of the music an individual has been exposed to over time, would it be

possible to build up an 'audible portrait' of an area, and how would this be best visualised / embodied?

- **Music distribution** – What might the impacts be on music distribution of a system of recommendation like this? Would the overall popularity of certain genres or artist be affected? How?

4.4.2 Obstacles

Quite apart from the future work suggested in the final chapter, a study of this size would most probably require additional testing and validation of the current code-base. Some of these additional technical challenges include:

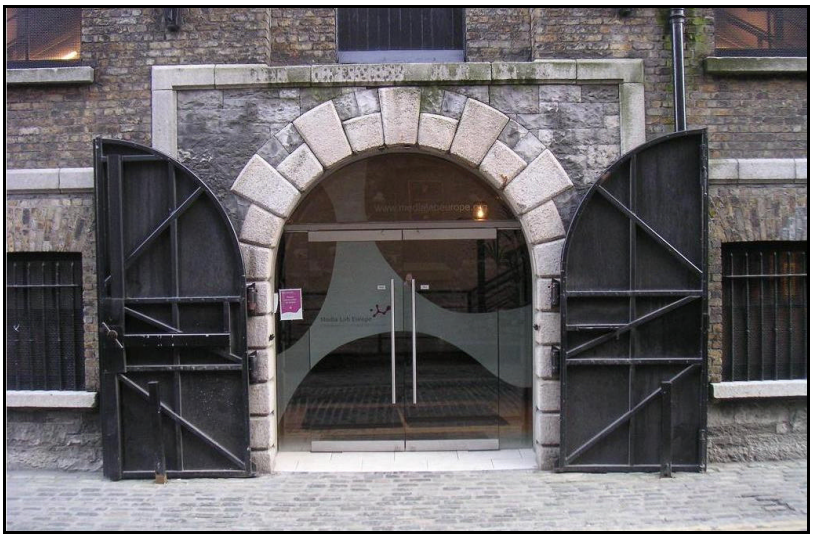
- **Backend Support** – To collect and analyse data from a hundred portable devices would necessitate exhaustive provisioning and planning. At a minimum, some form of merge/replication system would be required to upload all the data gathered into a central database. A tighter schema/nomenclature to identify individual devices and users would also need to be devised.
- **Bandwidth** – The 802.11b standard provides for a bandwidth of 11 Mbps (see Appendix H). Given that the audio streams of the type used by *tunA* require 112kbps for uninterrupted playback, a maximum of almost 100 simultaneous streams is theoretically possible, but highly unlikely given the poor performance of the networking hardware as seen to date. More extensive research is required to determine what the practical limits of such a system may be, and how other traffic in the same channel may affect these limits.
- **Packet loss** – UDP is at best an unreliable transmission protocol, yet given the conflicting needs of one-to-many communication, and high-packet throughput, some fine-tuning of the streaming process may be required. With the limited resources available on each device, it is not obvious which form this re-tuning may take.

4.5 Exhibitions / Installations

In addition to the data gathered through structured user studies, a great deal of useful feedback was obtained through other means, such as invited talks, publications, and extensive media coverage. Some of these are discussed below, and many more can be found in the Appendices.

4.5.1 Demonstrations

A permanent demonstration of *tunA* resides with the Human Connectedness group at Media Lab Europe. It consists of five h4155 iPaqs (see Appendix H) running the *tunA* application, each equipped with a memory card full of music. A variety of posters and papers relating to the project are also on display.



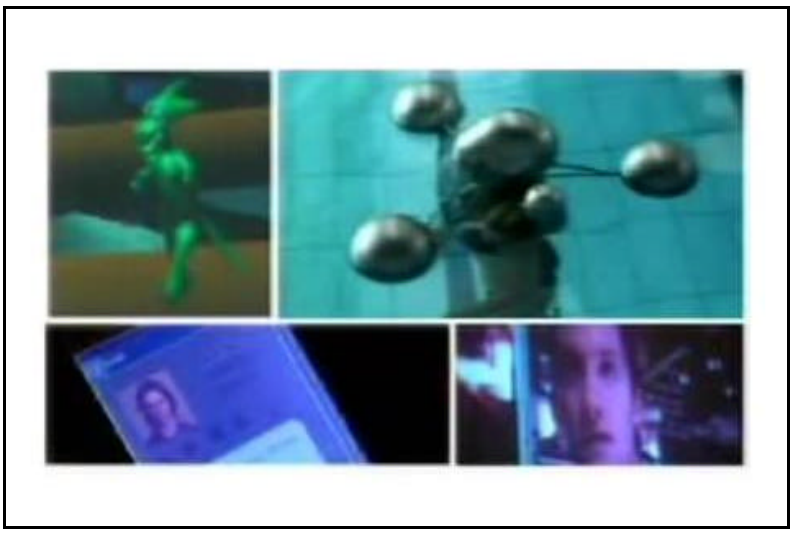
The front gates of MLE - Media Lab Europe, based in the old Guinness Hopstore, Dublin

Figure 4-16 Media Lab Europe – European Partner of MIT

All partners of both MITML and MLE have non-exclusive rights to commercialise the intellectual property generated by either lab. With this in mind, several current sponsors have expressed an interest in replicating the installation at their own R&D facilities.

4.5.2 Television / Radio

tunA also featured on the Network Two television show *Scope*, as part of an initiative by the Irish government to encourage more youngsters to chose a career in the sciences. RTE Radio subsequently aired a piece on *tunA* as part of its *Future Tense* technology broadcast. Both of these are supplied on the companion DVD-ROM.



Still from footage of the Network 2 TV show 'Scope', which featured a 'day-in-the-life-of' segment about the author

Figure 4-17 Video frame from *Scope* television show

4.5.3 Papers

The project also enjoyed success in the more academic sense, with the publication of several peer-reviewed papers at major conferences in Europe and the USA. Copies of the following are included as part of Appendix A

- *UbiComp 2003 - 5th Intl. Conf. on Ubiquitous Computing, Seattle, USA*
  "tunA: A Mobile Music Experience to Foster Local Interactions"
- *UbiComp 2004 - 6th Intl. Conf. on Ubiquitous Computing, UK*
  "tunA: Synchronised Music-Sharing on Handheld Devices"
- *WWC 2004 – 5th Wireless World Conference, Surrey, UK*
  "tunA: Local Music Sharing with Handheld Wi-Fi Devices"

4.5.4 Invited Talks

This media coverage, (particularly the *Wired* article – see Appendix D) generated a good deal of interest, which in turn begot invitations to several international conferences and workshops. For instance:

Sample slides from the CritDay presentation, see Appendix B for complete version

Figure 4-18 Samples slides from CritDay presentation

- *AMR 2004 – 3rd Austin Mobility Roundtable – Texas, USA*
  Arianna Bassoli was an invited speaker at this roundtable discussion
- *MMT 2004 – 1st Intl Workshop on Mobile Music – Goteborg, Sweden*
  tunA represented by a position paper and an invited speaker
- *MLE CritDay 2004 – Project Defence – Dublin, Ireland*
  A presentation of the project to a panel of experts from MIT, Imperial, UL, and UCD as part of a quality control exercise at MLE

4.5.5 Exhibitions

tunA was also exhibited both to the research community and to industry representatives at many stages during its evolution including:

- *ISEA 2004 – 12th Intl Symposium on Elec. Art – Helsinki, Finland*

Poster and demonstration to international art community

- *MLE Open_House 2004 – MLE Partner Events – Dublin, Ireland*

Live demonstrations given to many industry executives during a colloquium series hosted by the MediaLab as part of a sponsorship drive

4.6 Summary Remarks

This section presented the results of the initial questionnaires and pilot studies, the preliminary plans for a larger follow-up trial, and a summary of other related exhibitions and installations of *tunA* that also provided valuable feedback during the design cycle. From these, the following key points can be reliably inferred:

- **Compelling Concept** – The idea of being able to listen-in on the music of nearby strangers resonated with the majority of those who were exposed to it (see Section 4.2.2).
- **Clean Interface** – While the formal UI evaluation was somewhat ambiguous, the GUI performed admirably in the debriefing sessions.
- **Music as a Medium** – The passion it evokes in the majority of people makes music a perfect medium for a shared experience (almost half of those asked spend more than €400 a year on music).
- **Privacy a Factor** – Users are quite coy about the level of information they wish to share with strangers by default (less than one-in-four felt comfortable sharing even non-identifying data such as marital status).
- **Instant Messaging** – A recurring thread in the post-trial interviews was the need for built-in instant messaging functionality.
- **Location is Important** – There can be a substantial discrepancy between the locations where people are most curious about others, and those where they like to meet same (see Graphs 4-11/12).
- **Additional Work** – The need for a backend support system, and the potential issues surrounding bandwidth and packet loss mean extra testing and some modifications are pre-requisite to a larger rollout

In the final chapter the application itself is pitted against the original design goals derived earlier. Some obvious directions for future work and a last conclusion draw this thesis to a close.

5 Discussion / Conclusions

5.1 Closing Remarks

The aim of this work has been to develop a software platform for the exploration of new forms of social interaction, and music distribution.

The opening chapters of this thesis have examined the social imperative for such research, and through a comprehensive literary review have determined a number of goals to consider when developing such tools. This was followed by a discussion of the design and implementation of the *tunA* prototypes, and of several initial experiments involving their use.

This final section seeks to evaluate and reconcile the application itself against these earlier claims and goals. A brief comparison between the technical features of *tunA*, and the original aims that inspired them, is mingled with a summary of the results of the ethnographic studies to date. The chapter concludes with a series of directions for future work, and a few final thoughts.

5.2 Comparison with Design Goals

A grand total of sixteen design goals for personal technologies were compiled during the course of the second chapter, each of which influenced the design of the software platform. As an exercise in 'eating your own dog food'[27] the current prototype is now considered in the light of these goals (underlined).

tunA employs a synchronisation scheme to make to attempt to keep streams between multiple devices in time for a shared experience. It engages people with their environment by alerting them to the presence of nearby strangers and attempting to connect them through music. Rather than just anonymously sharing files, users are only permitted to 'listen-in' on tracks that are currently being player by others. Optimised for off-the-shelf PocketPC devices, *tunA* is already running on today's technology, and sports a flexible GUI which users can customise themselves. Playback is based on 112kpbs MPEG 1 Layer-3 stereo files, which support 'near CD-quality' fidelity. The open protocols used for transferring audio and data, mean that system can be subverted by advanced users although on a less positive note, this could be exploited to compromise user privacy. The system is accountable in that it logs all activity to a SQL database for later analysis, and functions as a standalone MP3 player in the absence of any other peers. Finally, brief exchanges between peers are possible over ad-hoc network connections, and a built-in instant messenger provides for bidirectional communication.

[27] 'Eating your own dog food' – A software engineering principle at Microsoft, where programmers working on future versions of the Windows O/S are forced to use their own beta code on their personal machines to encourage them to fix bugs in a timely fashion.

5.3 Future Work

This aside, there are several vectors for future work which could be explored. Several aspects of the software could be adapted or improved upon, others could be teased out into separate, more reusable modules, and the system could be extended to operate over different mediums or on different platforms – perhaps even a custom hardware implementation is in order.

5.3.1 Custom Hardware Platform

Some preliminary work was carried out on designing a custom hardware platform based around an Atmel DSP/microcontroller chip in conjunction with a Nordic VLSI 2.4 GHz RF module, and Secure Digital flash cards as a storage medium. This avenue was not explored extensively though, as it would require substantial hardware and software development, from a FAT16 file system for the flash media, to a multi-layer error-correcting radio stack. The only real advantage of this approach would be to lower the unit cost of the players, which is a secondary consideration in a purely research environment.

5.3.2 SmartPhone / Bluetooth

Another more obvious extension would be to rework the system to operate on multiple operating systems. The PocketPC platform has much in common with the more lightweight SmartPhone offering, (unsurprisingly as they are both based on Windows CE), and a port to this O/S would involve comparatively little effort. Modern Smart Phones such as the SPV range from Orange already ship with Windows Media player, and pack more than enough processing punch to handle decoding MP3 data in real-time. In a similar vein, support for additional networking mediums such as Bluetooth may increase market penetration were the system launched on a commercial level, and previous work involving streaming MP3 over Bluetooth (Gadd & Lenart, 2001) would suggest that is technically feasible. However, current Bluetooth implementations have a much shorter range than that of their WiFi counterparts (10m compared with 100m). In addition, the tendency seems to

be for newer devices to incorporate of all this functionality (the Motorola MPX is a prime example of this), which somewhat reduces the imperative of this particular avenue of future effort.

Figure 5-1 Motorola MPX, the next generation of mobile phone

5.3.3 Additional Media

For the *tunA* project, music was chosen as the shared media through which to encourage bonding, but there are many others that could be explored. Most mobile phones these days contain high-definition screens, and small video cameras, making shared video a possibility, likewise there exists an abundance of cheap off-the-shelf sensors which could be readily wired to a PDA/Phone to share heartbeats (Burke, 2004), biorhythms (Patel & Agamanolis, 2003), or any other imaginable space.

5.3.4 Multi-Hop / Hot Spots and Location Data

The current prototype is deliberately limited to single hop networks, in other words for two peers to 'see' each, they must be within range of each other. In a multi-hop system, if two peers are outside of each others range, but a third party exists in a 'piggy in the middle' position such that it can be seen by both, then the networking layer is smart enough to pass data from A to B, via C (see Figure 5-2 below). For this release of the software, it was felt that this would not be a particularly beneficial feature, as if two peers are out of range (in practice about 50m apart), then they are also likely to be out of sight, and no social interaction is likely to take place. However this functionality might

'come of age' if implemented in tandem with some additional hardware, such as a GPS receiver, so that some positioning information was available.

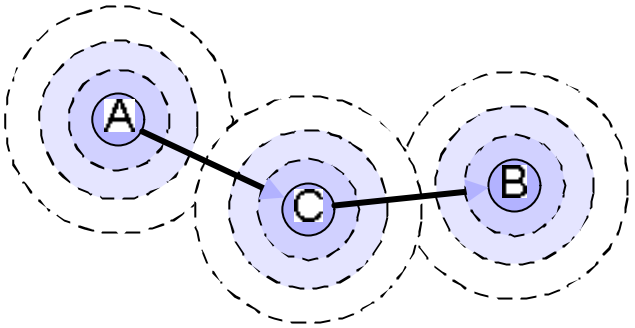


Figure 5-2 Message passing in a multi-hop network

5.3.5 Reusable Components

Several of the software systems could be spun-off into more reusable modules. As is, the C++ files can be readily ported into another application (already several stubs have been created which make use of parts of the tunA code base), however they could be wrapped up as WinCE services, or as DLL's to make them easier to use in other applications. In particular the networking components and custom graphical controls could be of great use.

5.3.6 Downloadable Release / Licensing

On a related note, it would be interesting to deploy the application on a much larger scale, more than the initial trials at the MediaLab and NCAD, and even on a larger scale than the proposed studies in the United States. One possibility would be to produce a version that could be downloaded from the web, and used on any compatible device. Intellectual property is the lifeblood of the MediaLab community, and there may be some issues in that area, but if the system were modified to periodically report anonymous usage data over the web, any number of interesting patterns may be observed.

5.4 Conclusion

This thesis began with a discussion of the damage that isolation can wreak on both a societal and an individual level. It was suggested that personal technologies in particular disenfranchised users from their environments; this and the exploration of new means of music distribution were offered as the primary justifications for the development of *tunA* – a 'socialising walkman', and tool for ethnographic study.

After a comprehensive literary review of related research in three areas (Mobile Audio, Social Networks, and Shared Experience), the main contribution of this work – the *tunA* application itself was discussed in detail. The novel features of this software (a custom algorithm for synchronising the playback of streaming media in ad-hoc networks, a proximity-based instant messenger, the ability to 'bookmark' favourite peers and audio, a novel user interface, and a bespoke input mechanism for touch screen mobile devices) were then explored with a mixture of diagrams, code extracts, and explanatory text. This application was then openly evaluated through the use of questionnaires, small field studies, formal user interface evaluations, and feedback from a number of exhibitions and papers.

Finally, a number of directions for future work were offered, including a custom hardware platform, various extensions to the existing code base, and some early plans for patenting, licensing, and distributing this technology.

References

Agamanolis, S. (2001). *Isis, Cabbage, and Viper: New tools and strategies for designing responsive media.* Unpublished PhD, Massachusetts Institute of Technology, Boston, MA USA.

Agamanolis, S. (2004). Designing Displays for Human Connectedness. In K. O'Hara, M. Perry, E. Churchill & D. Russell (Eds.), *Public and Situated Displays: Social and Interactional Aspects of Shared Display Technologies* (pp. 456). Dordrecht, The Netherlands: Kluwer Academic Publishers.

Aoki, P. M., Grinter, R. E., Hurst, A., Szymanski, M. H., Thornton, J. D., & Woodruff, A. (2002). Sotto Voce: Exploring the Interplay of Conversation and Mobile Audio Spaces. *Proceedings of CHI 2002 Conference on Human Factors in Computing Systems, Minneapolis, Minnesota, USA.*

Apple. (2004a). *iTunes - Share Music - Share Easily. Stream Wirelessly*, from http://www.apple.com/itunes/share.html

Apple. (2004b). *Rendezvous - Automatic Discovery of Computers, Devices, and Services on IP Networks*, from http://developer.apple.com/macosx/rendezvous/

Bach, E., Bygdas, S. S., Flydal-Blicfeldt, M., Mlonyeni, A., Myhre, O., Nyhus, S. I., et al. (2003). Bubbles: Navigation Multimedia Content in Mobile Ad-hoc Network. *Proceedings of the 2nd International Conference on Mobile and Ubiquitous Multimedia (MUM 2003), Norrkoping, Sweeden*, pp 73-80.

Barabasi, A.-L. (2002). *Linked: The New Science of Networks.* Cambridge, Mass.: Perseus Pub.

Bassoli, A., Moore, J., Cullinan, C., & Agamanolis, S. (2003). tunA: A Mobile Music Experience to Foster Local Interactions. *Proceedings of the Fifth International Conference on Ubiquitous Computing (UBICOMP 2003), Seatlle, WA USA.*

Bassoli, A., Quinn, I., Agius, M., Moran, A., Nisi, V., Dini, P., et al. (2002). Social Research for WAND and New Media Adoption on a Local Scale. *DYD 2nd International Conference on Open Collaborative Design for Sustainable Innovation, Bangalore*.

Bertsekas, D. P., & Gallager, R. G. (1992). *Data Networks* (2nd ed.). Englewood Cliffs, N.J.: Prentice Hall.

Blatt, B. (1985). Lessons from History. *Jounal of Learning Disabilities, 18*(1).

Boling, D. M. (2001). *Programming Microsoft Windows CE* (2nd ed.). Redmond, Wash.: Microsoft Press.

Borovoy, R., Martin, F., Resnick, M., & Silverman, B. (1998). GroupWear: Nametags that Tell About Relationships. *Proceedings of CHI 1998 Conference on Human Factors in Computing Systems, Los Angelas, CA USA.*

Borovoy, R., Martin, F., Vemuri, S., Resnick, M., Silverman, B., & Hancock, C. (1998). Meme Tags and Community Mirrors: Moving from Conferences to Collaboration. *Proceedings of the ACM 1998 Conference on Computer Supported Cooperative Work (CSCW 1998), Seattle, WA, USA.*

Boyd, D. M. (2004). Friendster and Publicly Articulated Social Networking. *Proceedings of Conference on Human Factors and Computing Systems (CHI 2004), Vienna, Austria*.

Brieger, R. L. (2002). The Analysis of Social Networks. In M. Hardy & A. Bryman (Eds.), *Handbook of Data Analysis*. California, USA: Sage Publications.

Brucker-Cohen, J., & Agamanolis, S. (2004). *Status Play - Adding Sociability to the iChat Status Line*, from http://www.mle.media.mit.edu/~jonah/projects/statusplay

Burke, R. (2004). *Biomelodics*, from http://www.mle.media.mit.edu/~rob/#Biomelodics

Chapman, D. (1998). *Visual C++ 6 in 21 Days*. Indianapolis, Indiana: SAMS Publishing.

Clark, A., Kantor, D., Negroponte, D., & Staicut, I. (2004). *Awaire and Aura, a Portable Networked MP3 Player*, from http://stage.itp.nyu.edu/designexpo/awaire/The_Awaire_Team/device.html

Cullinan, C., & Agamanolis, S. (2003). Reflexion: A Responsive Virtual Mirror for Interpersonal Communication. *ECSCW 2003 8th European Conference on Computer Supported Cooperative Work, Helsinki.*

Dobson, J. (1977). *Dare to Discipline*. Toronto: Bantam Books.

Du Gay, P. (1997). *Doing Cultural Studies: The Story of the Sony Walkman*. London ; Thousand Oaks Calif.: Sage in association with The Open University.

Eagle, N. (2004). Serendipity: Taking the Chance out of Chance Encounters. *Frames - An MIT MediaLab Sponsors Publication.*

Esbjornsson, M., Juhlin, O., & Ostergren, M. (2002). The Hocman Prototype - Fast Motor Bikers and Ad Hoc Networking. *Proceedings of the First International Conference on Mobile and Ubiquitous Multimedia (MUM 2002), Oulu, Finland.*

Esbjornsson, M., Juhlin, O., & Ostergren, M. (2003). Adding Value to Traffic Encounters: A Design Rationale for Mobile Ad Hoc Computing Services. *Proceedings of the 26th Information Systems Research Seminar in Scandinavia (IRIS 26), Haikko, Finland.*

Figueira, J. P. (20004). *QA: How do I embed a Pocket Word document in my application?*, from http://www.pocketpcdn.com/articles/word_embed.html

Finlayson, R. (1999). *Streaming MP3 via IP Multicast*, from http://www.live.com/mbone/mp3summit1999.ppt

Finlayson, R. (1999). *Streaming MP3 via IP Multicast*

Firelight-Technologies. (2004). *FMOD Music & Sound Effects System*, from http://www.fmod.org

Flickenger, R. (2002). *Building Wireless Community Networks* (First ed.). Sebsatopol, California: O'Reilly & Associates.

Gadd, S., & Lenart, T. (2001). *A Hardware accelerated MP3 Decoder with Bluetooth Streaming Capabilities.* Unpublished Masters, Lund University, Lund.

Gaye, L., Maze, R., & Holmquist, L. E. (2003). Sonic City: The Urban Environment as a Musical Interface. *Proceedings of the 2003 Conference on New Interfaces for Musical Expression (NIME-03), Montreal, Canada*, pp 109-115.

Grattan, N., & Brian, M. (2000). *Windows CE 3.0 Application Programming*: Prentice Hall PTR.

Haase, K. (2000). Why the Media Lab works - A personal view. *IBM Systems Journal, 39*(3&4), pp 419-431.

Hacker, S. (2000). *MP3 the Definitive Guide* (1st ed.). Sebastopol Calif.: O'Reilly.

Hawes, N. (2004). Commen Sense T9.

House, J., Landis, K. R., & Umberson, D. (1998). Social Relationships and Health. *Science, 241*(4865), pp. 540-545.

Isaacs, E., Walendowski, A., & Ranganthan, D. (2002). Hubbub: A sound-enhanced mobile instant messenger that supports awareness and opportunistic interactions. *Proceedings of CHI 2002 Conference on Human Factors in Computing Systems, Minneapolis, Minnesota, USA, 1*.

Jazzbo. (1999). *Lookin' For Love in New Cyber Spaces*, from http://www.geocities.com/Pentagon/Bunker/5921/lovegety.html

Laibowitz, M., & Paradiso, J. (2003). *SK1N Badge Design - A project proposal for the 2003 spring consortium badges* (Proposal). Boston, MA, USA: MIT Media Lab.

Leslie, R. (2004). *MAD: MPEG Audio Decoder*, from http://www.underbit.com/products/mad/

Lessig, L. (2001). *The Future of Ideas : The Fate of The Commons In a Connected World* (1st ed.). New York: Random House.

Lewis, Amini, & Lannon. (2003). *A General Theory of Love.*

Li, Q., & Rus, D. (2004). Global Clock Synchronization in Sensor Networks. *Proceedings of the IEEE, Infocom 2004*.

Liberty, J. (2001). *C++ in 21 Days* (First Edition ed.). Indianapolis, Indiana: SAMS Publishing.

Liu, J. W. S. (2000). *Real-Time Systems.* Upper Saddle River, N.J.: Prentice Hall.

Makofsky, S. (2004). *Pocket PC Network Programming*. Boston, MA: Addison-Wesley.

McCracken, G. (1988). *The Long Interview* (1st ed.). Newbury Park, California: SAGE Publications.

Menn, J. (2003). *All the Rave: The Rise and Fall of Shawn Fanning's Napster* (First ed.). New York, New York USA: Crown Business.

Mueller, F., Agamanolis, S., & Picard, R. (2003). Exertion Interfaces: Sports Over a Distance for Social Bonding and Fun. *Proceedings of CHI 2003 Conference on Human Factors in Computing Systems, Ft. Lauderdale*, pp 561 - 568.

Nemirovsky, P., & Davenport, G. (1999). Aesthetic Forms of Expression as Information Delivery Units. *Proceedings of CSNLP-8 The Eigth International Workshop on the Cognitive Science of Natural Language Processing, Galway Ireland*.

Neuros. (2004). *Neuros Audio FactSheet*, from http://www.neurosaudio.com/press/files/neuros_factsheet.pdf

Nielsen, J. (1994). *Usability Engineering*: Morgan Kaufman.

Nilsson, M. (2001). *ID3 tag version 2.4.0 - Main Structure*, from http://www.id3lib.org/id3/id3v2.4.0-structure.txt

Noll, P. (2000). *MPEG Digital Audio Coding Standards* (1 ed. Vol. 1). Berlin: CRC Press LLC.

Oram, A. (2001). *Peer-to-Peer: Harnessing the Power of Disruptive Technologies* (First ed.). Sebastopol, Calirfornia: O'Reilly & Associates.

Orlowski, A. (2004). *Prosmiscuous BluePod file swapping - coming to a PDS near you*, from http://www.theregister.co.uk/2004/06/03/pocket_rendezvous/

Painter, T., & Spanias, A. (2000). Perceptual Coding of Digital Audio. *Proceedings of the IEEE*, *88*(4), pp 451 513.

Papdakis, I., & Douligeris, C. (2001). Design and Architecture of a Digital Music Library on the Web. *Proceedings of the 3rd International Conference on Information Integration and Web-Based Applications & Services (IWAS 2001), Austria*, pp 445-447.

Patel, D., & Agamanolis, S. (2003). Habitat: Awareness of Life Rhythms Over a Distance Using Networked Furniture. *Extended Abstracts of*

UBICOMP 2003 - Fifth International Conference on Human Factors in Computing, Seattle, WA USA.

Perkins, C. E. (2001). *Ad Hoc Networking* (First Edition ed.). Uppper Saddle River, NJ: Addison-Wesley.

Petzold, C. (1999). *Programming Windows* (5th ed.). Redmond, Wash.: Microsoft Press.

Pistorius, P. (1981). *Kind in Krisis*. Cape Town: Human & Rousseau.

Putnam, R. D. (2000). *Bowling alone : the collapse and revival of American community*. New York: Simon & Schuster.

Rodenstein, R., & Donath, J. S. (2000). Talking in Circles: Designing A Spatially-Grounded Audioconferencing Environment. *Proceedings of CHI 1999 Conference on Human Factors in Computing Systems, Pennsylvania, PA USA*, pp 81-88.

Sanneblad, J. (2004). *GapiDraw - The Graphics SDK for Stationary PC's, SmartPhones, and Handheld Computers*, 2004, from http://www.develant.com/gapidraw.asp

Sanneblad, J., & Holmquist, L. E. (2004). OpenTrek: A Platform for Developing Interactive Networked Games on Mobile Devices. *Proceedings of the Fifth International Symposium on Human Computer Interaction with Mobile Devices and Services (Mobile HCI 2003), Udine, Itlay*.

Sartwell, C. (1999). *Walking Alone*, from http://www.crispinsartwell.com/media/walkman.htm

SharmanNetworks. (2004). *Kazaa Media Desktop - A Peer to Peer File Sharing Application*, from http://www.kazaa.com/us/index.htm

Shockfish. (2004). *Spot Me - The Know-Who for Meeting People*, from http://www.spotme.ch/spotmeinfo.html

Sklar, B. (2001). *Digital communications : fundamentals and applications* (2nd ed.). Upper Saddle River, N.J.: Prentice-Hall PTR.

Sony. (2004). *History of the Sony Walkman*, from http://www.sony.net/Fun/SH/1-17/h2.html

Stien, P. T., & Kendall, J. C. (2004). *Psychological trauma and the developing brain : neurologically based interventions for troubled children*. New York: Haworth Maltreatment and Trauma Press.

Tanaka, A. (2004). Mobile Music Making. *Proceedings of the 2004 Conference on New Interfaces for Musical Expression (NIME04), Hamamatsu, Japan*, pp154-156.

Tanenbaum, A. S. (2003). *Computer networks* (4th ed.). Upper Saddle River, NJ: Prentice Hall PTR.

Tanenbaum, A. S., & Steen, M. v. (2002). *Distributed Systems : Principles and Paradigms*. Upper Saddle River, N.J.: Prentice Hall.

Terdiman, D. (2004, Mar 22 2004). Brits Going at It Tooth and Nail. *Wired Magazine*.

TunesAtWork. (2004). *Tunes at Work - Leave Your Music at Home, and Listen at Work. Free.*, from http://www.tunesatwork.com/

Wimpy. (2004). *Wimpy - A Macromedia Flash Based Streaming Player and Streaming Server Solution*, from http://www.wimpyplayer.com/

Winder, R. (1991). *Developing C++ software*. Chichester, England: J. Wiley.

Glossary

.DLL	Dynamic Link Library
.LIB	Static library file containing exposed functions and entry points
.SKN	*tunA* skin files, for customising the user interface
802.11	Ubiquitous Wireless networking standard
BMP	Bitmap
CBR	Continual Bit Rate
FMOD	Multi-platform music library available at www.fmod.org
GIF	Graphics Interchange Format – Compressed image file format
ID3	Metadata sometimes found in MP3 files
LAN	Local Area Network
MFC	Microsoft Foundation Classes
MP3	MPEG Version 1 Layer 3 – Popular audio codec
UML	Unified Modelling Language
VBR	Variable Bit Rate
WIFI	Alternative name for 802.11
WLAN	Wireless Local Area network

tunA: Shared Audio Experience

Julian Moore

This thesis presents *tunA* – a 'socialising MP3 player'. An evolution of the walkman, it is a peer-to-peer software application for mobile devices that as well as functioning as a normal MP3 player, allows users to locate nearby peers, view their profiles, send instant messages, and wirelessly share music. Rather than simply exchanging music libraries, the software streams audio from device to device in a tightly synchronised fashion, so that all parties hear the same audio, at the same time. This combination of discovery mechanism, profile and message swapping, and 'shared audio experience' provide a platform for people to encounter and engage with others of similar tastes and interests. A comprehensive logging mechanism completes the application, and extends it into a valuable tool for the study of social dynamics.

Technically, *tunA* sports a number of unique characteristics, several of which are the subject of a provisional United States patent. These features include: a custom algorithm for synchronising the playback of streaming media in ad-hoc networks, a proximity-based instant messenger, the ability to 'bookmark' favourite peers / audio, and a novel user interface (including a bespoke input mechanism for touch screen mobile devices).

Ethnographically, *tunA* aims to uncover and explore new forms of social interaction, evolution of musical tastes, audio portraits, listening habits, and music distribution.